Knowledge Mining over Scientific Literature and Technical Documentation

Thesis

presented to the Faculty of Arts

of the University of Zurich

for the degree of Doctor of Philosophy

by

Fabio Rinaldi

from Italy

Accepted in the spring semester 2008

on the recommendation of

Prof. Dr. Michael Hess

and Prof. Margaret King

Zurich, 2009

Contents

Acknowledgments

The work described in this dissertation would not have been possible without the collaboration, support and friendship of numerous people who provided help and inspiration over the years.

First of all, the colleagues with whom I have most closely cooperated: James Dowdall, Michael Hess, Diego Molla, Kaarel Kaljurand, Gerold Schneider, Rolf Schwitter.

Other colleagues who have helped with their friendships and occasional suggestions include: Maya Bangerter, Alexandra Bünzli, Simon Clematide, Norbert Fuchs, Manfred Klenner, Cerstin Mahlow, Martin Volk.

Other colleagues with whom I have collaborated in papers and projects include: Bill Black, Eric and Fidelia Ibekwe-Sanjuan, John McNaught, William Lowe, Andreas Persidis.

I owe a lot to the personal support and patience of my family and friends, including in particular my parents, my sister, and my life partner of many years, Elia Yuste.

Finally I would like to thank my supervisors: Michael Hess, for always granting me a large amount of intellectual freedom, and Maghi King, whose support in the final phases proved to be invaluable.

Abstract

This dissertation is based upon research activities carried out from 2000 onwards, initially in the framework of an existing internal research project (ExtrAns, 2000-2002), later partially within the scope of a European project (Parmenides, 2003-2005) and then in the context of a second internal research project (OntoGene, 2005-present).

The common focus of all these activities has been an investigation of how to make use of linguistic items closely tied to the specific domain of discourse (technical terminology and domain specific collocations) to improve automatic extraction of information from textual sources (and in particular from technical and scientific documents). I have deliberately avoided using the term 'Information Extraction' in the title, as I believe that this term has assumed (in scientific discourse) a rather narrow connotation, which does not cover completely the research activities reported here.

In the following I will first describe the core research question that this work tries to answer and then provide a brief outline of the contents of the dissertation.

Motivation

The wealth of information that modern society offers to its members is at the same time a blessing and a problem. Internet users can access a mass of knowledge that was unimaginable only two decades ago. Information relevant to a specific problem had to be painstakingly searched through traditional libraries. Limited information retrieval interfaces allowed better

access, but only to specific document collections.

In just a little more than a decade, the information landscape has completely changed. This process has involved technological advances, economical progress and societal changes. The growth of the Internet as a means of distributing documents and multimedia information has radically altered our approach to information. At the same time, cheaper computing hardware has allowed almost every household to purchase the equipment needed for access. Today's motto is *"all the world's information at your fingertips"*.

However, abundance of information does not necessarily translate into a better capacity to use such information. Information has to be distilled into knowledge before it can be acted upon. While the problem of information distribution and access can be considered solved to a large extent,[1] the problem of finding the right information at the right time, and in a format that is easy to assimilate, is still an open research question.

The transformation of information (in particular that contained in textual documents) into formalized knowledge (electronically stored and mechanically processable) can be seen as the main focus of this work. In order to be used for practical purposes, the knowledge that can be extracted needs to be made accessible through suitable user interfaces. Although different modalities of access can be considered (e.g. visual representation), we will focus exclusively on modalities that make use of some form of direct user query, expressed in a textual format, and requiring as output again a textual format: in other words, written requests for information expressed in a natural language and system generated responses also expressed in written natural language. Even with this restriction on input and output modalities, the overall goal remains very ambitious, and can be only partially fulfilled

[1] Although different languages still constitute a mighty barrier, which can partially be overcome by new generation machine translation systems.

within the scope of a doctoral dissertation. We will focus in particular on making use of the domain knowledge which is implicitly encoded in technical terms and domain specific collocations (**domain descriptions**, in the parlance of this thesis), within the scope of a restricted-domain question answering system.

Outline

The process of transformation of textual documents into whatever formal representation has been chosen is based upon various techniques from the field of natural language processing, none of which is at present capable of delivering perfect results, so that some amount of error is unavoidable. The aim of our work is to reduce the degree of error and at the same time offer to the end-users a system which can be employed in a real-world context, to increase their productivity by reducing the time required for finding relevant information.

The traditional research area of "Question Answering" (QA) tries to provide a partial solution to this problem by offering users a chance to find relevant information from a document collection using queries expressed in natural language. Since 1999 activities of this sort have been promoted by a specialized track within the US-funded TREC competition. As background, we provide a detailed survey of such research efforts.

While most of the QA systems have been developed as "open-domain" systems, their chances of finding a relevant response to a question are always limited by the document collection that they use as their background knowledge source. Furthermore, their "openness", while desirable in principle, is also a limitation. In order to be as general as possible, given that current NLP techniques are fraught with errors, and that many of the resources

needed for deep understanding are not available, they are forced to fall back on ad-hoc solutions, which in part restrict their intellectual contribution to the challenge of achieving some form of machine understanding.

We have chosen to follow a different path, and have focused instead on "restricted domain" question answering, in particular on technical documentation and scientific literature. The choice of a specific domain allows us to make use of specialized resources, in particular terminologies and ontologies, which in some cases are already available. The problem of semantically grounding the entities of discourse can then, at least in part, be solved by making use of such resources. For examples, all parts of an aircraft, although they might be named using different names, will have a unique identifier, which can be taken as their referent, and can be traced within standard databases. In a similar fashion, many elements of scientific discourse, in particular in the biomedical field, are increasingly given official identifiers in standard resources. A good example is UniProt, a comprehensive catalog of information on proteins.

Where no terminology resources are already available, they need to be constructed. An early part of the research reported here was concerned with evaluating the potential utility of existing term extraction tools and with the construction of a list of domain specific linguistic items for use in experimentation. In accordance with our general hypothesis, these linguistic items cover both technical terminology in the strict sense and domain specific collocations.

However, even technical terminology in the strictest sense shows a degree of variability which is often underestimated and which poses challenges in its own right, challenges not usually dealt with by open-domain QA systems. Since different (but linguistically related) terms can be used to refer to the same domain entity, it is necessary to recognize all possible variants of a

given term and possibly structure them so that they can be used in the question answering process. We deal in detail with such problems, and present the solution adopted in an extension to an existing system (ExtrAns), aimed at the technical maintenance manual of a well-known commercial aircraft. ExtrAns is a fully functional answer extraction system (developed before the start of this dissertation) using the "unix manpages" as document collection, and allowing flexible question answering over the domain of unix commands. The original version of ExtrAns did not make use of domain descriptions, which are the novelty introduced by the present work.
The explicit treatment of domain descriptions provided considerable gains in terms of efficiency, in particular in the process of analysis of the background document collection, and (to a more limited extent) in the processing of the user queries.
The results achieved with the extension to the ExtrAns system provided the basis for our more recent research activities, which moved into the field of biomedical scientific literature, focusing in particular on protein-protein interactions. We have built a system which is capable of extracting, with good precision, interactions of this type (a working prototype can be accessed online). We have recently participated in a competitive evaluation aimed at tackling this problem, where our system has obtained good results.
This dissertation describes the research activities sketched above, and in particular:

- The extraction of domain specific lexical items (which we will call "domain descriptions") which refer to entities of the domain, and are composed in part by traditional terminology and in part by domain specific collocations.

- The detection of relationships (like synonymy and hyponymy) among

the domain descriptions, and their organization into a conceptual structure.

- The usage of the domain descriptions and their relationships within a specific domain restricted question answering system, in order to facilitate the correct identification of a relevant answer to a query.
- The adaptation of the system to another domain, and extension of the basic hypothesis to tasks other than question answering.

The work described in this dissertation is the result of research activities carried out by the author as coordinator of the WebExtrAns project (2000 - 2004) and as team leader within the scope of the EU FP5 project PARMENIDES in the years 2003 to 2005. To ensure their timely dissemination, some of the research results reported in this work have previously appeared in various scientific publications, including [Rinaldi et al., 2002c, Rinaldi et al., 2002b, Rinaldi et al., 2002a, Rinaldi et al., 2003b, Mollá et al., 2003b, Mollá et al., 2003a, Rinaldi et al., 2003d, Rinaldi et al., 2003c, Rinaldi and Yuste, 2003, Dowdall et al., 2004, Rinaldi et al., 2004a, Rinaldi et al., 2004b, Rinaldi et al., 2005].

Chapter 1

Introduction

The wealth of information that modern society offers to its members is at the same time a blessing and a problem. Internet users can access a mass of knowledge that was unimaginable only two decades ago. Information relevant to a specific problem had to be painstakingly searched through traditional libraries. Limited information retrieval interfaces allowed better access, but only to restricted document collections.

In only a little more than a decade, the information landscape has completely changed. This process has involved technological advances, economical progress and societal changes. The diffusion of the internet as a means of distributing documents and multimedia information has radically altered our approach to information. At the same time, cheaper computing hardware has allowed almost every household to purchase the equipment needed for access. Today's motto is *"all the world's information at your fingertips"*.

However, abundance of information does not necessarily translate into better capacity to use such information. Information has to be distilled into knowledge, before it can be acted upon. While the problem of information distribution and access can be considered solved to a large extent (at least

when multilingual/multimedia issues are ignored), the problem of finding the right information at the right time, and in a format that is easy to assimilate, is still an open research question.

The transformation of information (in particular textual documents) into formalized knowledge (electronically stored and mechanically processable) can be seen as the main focus of this work. Such formalized knowledge, in order to be used for practical purposes, needs to be made accessible through suitable user interfaces. While different modalities of access can be considered (e.g. visual representation), we will focus exclusively on modalities that make use of some form of direct user query, expressed in a textual format, and requiring as output again a textual format. Even with this restriction, the goal of finding and presenting the right information is still very ambitious and, in order to be even partially fulfilled, requires contributions from a number of different areas of science.

The definition of "knowledge" is a philosophical problem that goes well beyond the scope of this work. However, even for our practical aims, one cannot proceed without making choices on *what* should be represented (what are the conceptual entities that need to be represented), and *how* it should be represented (which representation format should be adopted). Solutions to the first problem have been proposed in the form of so-called ontologies. The second problem is the focus of the knowledge representation community, which has proposed a number of different formalisms that could be used in a computational system.

Information Retrieval (IR) techniques, originally developed for retrieving documents about a specific topic among a large document collection, have been adopted as a way to navigate through the Web, seen as a huge document collection. While the traditional algorithms proved successful to some extent, new algorithms have been developed that exploit the interlinked na-

ture of web documents (e.g. PageRank, the algorithm used by Google).
Still, the basic assumption of the IR approach, namely the retrieval of full documents (possibly ranked in order of relevance) in response to a given query, has not been challenged. There are two major problems with this paradigm. First, the user is still left with the task of reading the retrieved documents in order to decide if they are relevant or not. Second, the process of distilling the relevant information into useful knowledge is entirely left to the user.

In this dissertation we explore various paradigms of information access to document collections, mainly of a technical nature. One possible approach to solve these problems is called "Question Answering" (QA). In the QA paradigm, the user can enter a question in a natural format (i.e. using his/her own language) and the system provides a direct answer to that question. We will assume that the question is provided to the system in written format, although in practice it might be possible to use a speech-to-text interface to do away with the need to type the question. Question answering is an active area of research at present, because of its great potential to solve the information bottleneck problem: provide quick and focused access only to the knowledge needed to solve a specific problem. Commercial systems that make use of QA interfaces do already exist, however they have many shortcomings that the research community is addressing. A well-known example is the service formerly known as AskJeeves,[1] which however is only superficially a QA system. There are also a few research QA systems which are made accessible through web interfaces. Examples are AnswerBus[2] and Start.[3]

The current research paradigm assumes that the knowledge from which the

[1] <http://www.ask.com/>
[2] <http://www.answerbus.com/index.shtml>
[3] <http://start.csail.mit.edu/>

system derives the answers is not coded in a structured repository (e.g. a knowledge base) prior to the formulation of the question, but is instead derived by the system itself, using a text collection as a source of information. This approach has been promulgated in particular in the TREC question answering competitions (see Section 2.3), where participant systems are given a large document collection[4] from which answers have to be retrieved.

In some recent applications, the entire web is taken as the background document collection, so systems are free to find their answers on any web page. The information available on the web is massively redundant, thus creating a potential for considerable simplification of the amount of linguistic processing required to retrieve the information sought for: as the same facts are likely to appear in different formulations, some of those formulations might be so close to the original question that they will be relatively easy to identify correctly (see Section 1.2 for more detailed discussions and examples).

While the TREC competitions have focused exclusively on the English language, interest in multilingual QA is growing, and a separate evaluation (Multiple Language Question Answering, see <http://clef-qa.itc.it/>) has been set up within the framework of CLEF (Cross Language Evaluation Forum): <http://www.clef-campaign.org/>.

We provide at this point a short glossary of the terminology most frequently used when discussing Question Answering Systems:

- **Document Collection:** the reference collection that is to be searched in order to locate answers to the questions. Different types of collections are considered, varying in size and genre. From organizational documents, to technical manuals or newswire reports, up to the entire internet.

[4] ca. 1,000,000 documents / 3 GB, all newspaper/newswire, mostly American

- **Question type:** categorization of question for purposes of distinguishing between different processing strategies and/or answer formats. The reason for such a categorization is that different types of questions require the use of different strategies to find the answer. A large variety of different question types is tackled by current QA research: factoid, list, definition, how, why, hypothetical, semantically-constrained, and cross-lingual questions. There are different ways to classify questions into specific classes, and some systems make use of a refined taxonomy.

- **Answer type:** class of objects involved by the question (e.g. persons, organizations, locations, dates, and so on). Generally such objects are related to the named entities (NE) identified into the question.

- **Question focus:** property/entity in which the question is interested

- **Question topic:** object/event the question is about

  E.g. In the question *"What is the length of the Mississippi?"*, the focus is the length, while the topic is the Mississippi.

- **Candidate Passage:** a text fragment of any length (as short as a sentence, or the entire document) retrieved by a search engine in response to a query generated on the basis of the user's question.

- **Candidate Answer:** a short fragment (typically a few words) from a candidate passage, which is considered by the QA system as a potential answer to the question.

- **Answer Extraction:** in much of the literature on question answering systems, this term is taken to refer to a specific stage of processing, involving in particular the identification of the text snippet(s) to be de-

livered to the end user. However, it has been claimed many times that, because current question answering systems do not generate their answers, but simply return a text snippet, it would be more appropriate to call them "Answer Extraction systems", and reserve the term "question answering" for more powerful systems, making use of deductive capabilities and capable of generating answers not directly contained in the documents, but constructed on the basis of information inferred from them.

While in some of our previous work [Rinaldi et al., 2002c, Hess et al., 2002] we used preferentially the term 'Answer Extraction Systems' for the TREC-type systems, in this dissertation, unless explicitly noted, we will make use of the established terminology.

Consider as an example the questions: *"Where was Albert Einstein born?"* The answer could be found in many places, for instance a book containing biographies of famous scientists (which would then be our *document collection*). The question is asking for a particular fact, and can therefore be classified as a factoid question (*question type*). The answer required is a particular place, the *answer type* is therefore a location. The name of the town could be a good answer: in this case a town would be the question focus.[5] The *question topic* is Albert Einstein. A *candidate passage* would be the following: *"The house in Ulm where Einstein was born. Soon afterwards the family moved to Munich, a bustling city where his father hoped to find a better environment for his shaky business"*.[6] A *candidate answer* would be just the string *"Ulm"* from the passage above.

[5] Notice however that the question is underspecified, so what exactly constitutes a good answer (a country, a town, or a complete address) depends on the system's interpretation of the user's knowledge state.

[6] <http://www.aip.org/history/einstein/ae1.htm>

Typically QA systems are distinguished into 'Closed-domain'[7] and 'Open-domain' systems. While the former deal with questions in a specific domain (e.g. technical manuals, biomedicine), the latter are supposed to deal with generic questions. Closed-domain QA systems can make use of domain-specific knowledge, like ontologies or terminologies. On the other hand, open-domain systems can normally rely on very large document collections, which offer different opportunities to find the correct answer thus exploiting the redundancy inherent in the data.

This work describes in general the problem of question answering, and focuses in particular on restricted-domain QA systems. Our main claim is that a proper treatment of domain descriptions (terminology and domain specific collocations) is essential for QA in technical domains: without it, deep QA has severe limitations.

1.1 Background

In recent years, the information overload caused by the new media has made the shortcomings of traditional Information Retrieval increasingly evident. Practical needs of industry, government organizations and individual users alike push the research community towards systems that can exactly pinpoint those parts of documents that contain the information requested, rather than return a set of (perhaps) relevant documents. Question Answering (QA) systems aim to satisfy this need.

Traditional information retrieval (IR) techniques provide a very useful solution to a classical type of *information need*, which can be described with the scenario of 'Essay Writing'. The user needs to find some information and backup material on a particular topic, and she will sift through a number of

[7] Alternatively called 'restricted-domain QA systems'.

documents returned by the IR system. This assumes that the user has sufficient time to elaborate and extract the relevant information from a number of documents.[8] However, a different type of information need is becoming increasingly more common, namely one where the user has to solve a specific problem in a restricted domain, which requires finding precise information of a limited size. This could be called a 'Problem Solving' scenario. A fitting example is that of technical manuals. Imagine the situation of an airplane maintenance technician who needs to operate on a defective component which is preventing an airplane from starting. He needs to swiftly locate in the maintenance manual the specific procedure to replace that component. What users need in this situation are systems capable of analyzing a question (phrased in natural language) and searching for a precise answer in document collections.

Therefore some sections of the research community have focused their interest on systems which can not only locate relevant documents, but also pinpoint the exact piece of information that the user is interested in. In the 1990s, the Message Understanding Conferences (MUC) [Chinchor, 1998a] have been a major arena for development in this field. The concept of information extraction has been gradually developed and refined so that today this is considered a separate and autonomous area of research. Typically such systems can extract specific types of information predefined by the creators of the system. The simpler applications, like named entity extraction, have enjoyed considerable success. More complex applications, like template extraction and scenario extraction did not seem capable of improving significantly after reaching levels which were deemed interesting but not fully satisfactory. A fundamental problem with information extraction ap-

[8] It has been often observed that traditional Information Retrieval should rather be called 'Document Retrieval'.

plications of the complex type (template extraction, scenario extraction) is that the system is normally tailored to templates which have necessarily been predefined within the system. Thus adapting the system to a new domain requires new resources: the system is rigidly tied to a domain. A change in the specific interests of the user poses therefore major adaptation problems.[9]

Research in the area of question answering has been promoted in the past couple of years by, in particular, the QA track of the TREC competitions [Voorhees, 2000a, Voorhees and Harman, 2001]. The participants in this competition have the opportunity to measure how well their systems can retrieve answers to a predefined set of questions from a very large collection of documents. They run their system on the given questions and return for each a ranked list of five answers in the form of pairs [document identifier, answer string]. The returned data are then evaluated by human assessors, who for each string have to decide whether it contains an answer to the question and whether the given document supports that answer.

There are different levels of performance that can be expected from a question answering system, and a classification is not easy. However, a first broad distinction can be made on the basis of the type of knowledge that the system employs, which ultimately determines which questions the system can answer.

An ideal system would return a grammatically well-formed surface string generated from a non-linguistic knowledge base in response to a natural language query. Unfortunately, many problems in the Knowledge Repre-

[9]Some systems, based on machine learning, can be easily adapted to a different domain, if sufficient training data is provided. However their capabilities are constrained by the definition of the templates, and they could not possibly handle arbitrarily phrased questions.

sentation field are still to be solved and a comprehensive repository of world knowledge is not available.[10] What is achievable are systems that acquire their knowledge only from the target data (the documents to be queried). Such a system may allow inferences at the local/linguistic level or across multiple or single texts, depending on the task at hand.

The complexity of a Question Answering system could be defined in terms of the number and type of transformations that it can apply to the user query in order to match it with the answer. The most simple approach would be to allow only syntactic variants (such as active/passive), while more sophisticated approaches would gradually include detection of synonyms and of more complex lexical relations among words such as thesaurus relationships like *'subdirectory* is a subtype of *directory'* as well as textual references (pronouns, definite noun phrases), and finally the use of meaning postulates (such as *'if something is* installed *in some place, then it* is *there'*).

The focus of the TREC competitions has been predominantly factual (non-generic, extensional) questions about events, geography and history, such as *"When was Yemen reunified?"* or *"Who is the president of Ghana?"*. It has been observed repeatedly that many such questions would better be directed at encyclopedias rather than at newspaper articles. Questions concerning rule-like or definitional knowledge (generic, intensional questions), such as *"How do you stop a Diesel engine?"* or *"What is a typhoon?"* have received less attention.[11]

Trying to focus on QA systems over large volumes of data may leave aside an important range of applications. There will always be a need for technical documentation, and there will always be a need for tools that help people

[10] Despite some commendable efforts in this direction [Lenat, 1995].

[11] Although a small number of them were included in the QA track of TREC-9 and TREC-10.

find the information they want from technical documentations. A Linux user may want to know how to set a symbolic link to a file or a directory. A user of Photoshop may want to know how to improve the tonal range of an image. A member from an Airbus technical maintenance crew may want to know the location of the Electronic Centralised Aircraft Monitor contactor. These technical documentations are not large when compared with the data used in the TREC Question Answering track, and the user is unlikely to find the answer to some of these technical questions on the Web.

Approaches that rely on data redundancy do not work well in these domains for two reasons. First of all, the amount of text is not large enough and therefore problems of sparse data are likely to occur. Second, authors of technical manuals typically try to avoid redundancy of information, they do not want to explain the same concept more than once or twice. Trying to use data redundancy approaches in non-redundant data is a self-defeating task.

The formal writing in technical documentation makes it possible to write a grammar that will cover these texts. In fact, in an evaluation up to 90% of the sentences in a software manual were parsed by the publicly-available Link Grammar parsing system after incorporating specific lexical items [Sutcliffe and McElligott, 1996a]. Current parsing systems have improved since. It is therefore possible to build the logical form of a sentence and use it in the question answering process.

Given the non-redundant nature of technical texts, an approach that attempts to find the meaning of the text and use it for question answering can lead to more relevant results, compared to approaches that use bags of words or collections of sentence patterns. In other words, technical texts allow and require the use of NLP-intensive approaches.

If question answering is to perform satisfactorily in technical domains over

limited amounts of textual data with little redundancy it must make maximal use of the information contained in the documents. This means that the meaning of both queries and documents must be taken into account, by syntactic and semantic analysis. Our own QA system, ExtrAns, shows that such applications are within the reach of present-day technology.

Question-answering (QA) over technical domains is distinctly different from TREC-based QA or Web-based QA as it cannot benefit from data-intensive approaches. Technical questions arise in situations where concrete problems require specific answers and explanations. Finding a justification of the answer in the context of the document is essential if we have to solve a real-world problem. We use an existing question answering sytem (ExtrAns) to show that NLP techniques can be used successfully in technical domains for high-precision access to information stored in documents. We discuss an answer extraction system over technical domains, its architecture, its use of logical forms for answer extractions and how proper handling of domain descriptions, composed of domain specific collocations and terminology proper, becomes a crucial factor in the proper functioning of the system.

The specific treatment of domain descriptions is the main focus of the work presented in this dissertation. Their importance is motivated by the fact that they make up a high proportion of the lexical material within technical documents. Domain descriptions are typically MWEs with many possible syntactic structures, of which however one only is intended. Therefore a compilation of such units once and for all can bring various benefits to the analysis of technical documentation, as we will show.

Problem: **Domain descriptions are one of the key obstacles in processing technical documentation. They cause problems both at the syntactic and at the semantic level. Syntactically, domain descriptions are in**

general difficult to parse correctly: not only do they present all the standard problems of analysis common to multi-word expressions, but there is also an increased probability of finding unknown words or words used in a different way to their use in general language. This is bound to affect the parsing process, as the parser might attempt to create spurious syntactic constituents involving fractions of the term.[12] In other words, multi words expressions behave as a single syntactic unit which should not be further decomposed, and need to be treated as such in the parsing process. Semantically, an unambiguous compositional representation of their meaning might not always be attainable.

Solution: **We handle domain descriptions in a separate process from the handling of the rest of the document. Descriptions are initially isolated and organized into a semantic structure, which is later integrated into the processing of documents and queries.**

1.2 Methodologies

A number of different technologies have been used by QA systems. The initial problem to be solved is that of locating the interesting passages and sentences from the retrieved documents. Typically keyword-based techniques are used, where the keywords are derived from the initial question. The candidate passages or sentences are then filtered, possibly based on the pres-

[12]Different parsers might have different strategies to deal with unknown words. Some parsers might simply ignore them, leading to partial parses or complete parsing failure. Other parsers might attempt to 'guess' the syntactic category of unknown words, in order to recover from a potentially fatal situation. In this case, they might assign to the unknown word any of the 'open class' categories, which might lead to spurious parses. Obviously, the parsing problems created by unknown words are the same, whether they occur inside domain descriptions or not.

ence of a specific answer type in the text. Various features (syntactic, lexical, positional, etc.) are then used to rank the candidate answers.

Very large document collections (e.g. the Web) offer various forms of data redundancy. They are likely to contain the required answer phrased in different ways, possibly in different contexts, thus increasing the chances of locating it. In such cases it is possible to use simple templates to find the answer to many question types. If the question is *"What is aquaporin?"*, a system should be able to identify the pattern *"What is an X"* and look for documents which match the pattern *"X is a Y"*. This often works well on simple 'factoid' questions seeking factual bits of information such as names, dates, locations, and quantities.

In many other cases such simple reformulation might not be sufficient, either because of the complexity of the question, or because the background collection offers less redundancy. Systems have to resort to more complex techniques, which might involve syntactic, semantic and contextual processing of the question and of the candidate answers.

An important methodological problem is to establish what constitutes a good answer, or to what degree a given answer satisfies a given question. In many cases the quality of an answer can be judged only in respect to the previous knowledge and intentions of the person asking the question. For example, for the lay person, the text *"a protein"* might be a perfectly reasonable answer to the question *"What is aquaporin?"*, however for the biologist a good answer should include a brief description of the function of the protein.[13]

There are different degrees in which a good answer matches a given question. They might be (almost) identical on the surface (word) level. They

[13] Aquaporin is a protein located in the cell outer membrane, which is responsible for the passage of water molecules from the outside environment to the inside of the cell.

might be lexically similar, but involve some syntactic variation. They might be similar modulo replacement of synonyms. In order to cope with all these possible variations, different resources are needed, which provide linguistic information (surface, lexical, syntactic, semantic, pragmatic) and ontological (and common sense) information: thesauri, domain ontologies, lexical semantics, common sense repositories, etc.

The concept of "Textual Entailment" has been proposed as a way to formally connect questions and corresponding answers, as a solution for modelling language variability in different NLP tasks. The formal definition of textual entailment is as a relationship between a text T and an expression H (called the hypothesis): "T is said to H (T $\longrightarrow$ H) if the meaning of H can be inferred from the meaning of T. An entailment function e(T,H) thus maps an entailment pair T-H to a truth value. Alternatively, e(T,H) can also be intended as a probabilistic function mapping the pair T-H to a real value between 0 and 1, expressing the confidence that a human judge or an automatic system estimates the relationship to hold."[14] A simple case of entailment is taxonomic entailment: *"A cat exists"* $\longrightarrow$ *"An animal exists"*. Another case is synonym-based paraphrasing: *"A purchased B"* $\longrightarrow$ *"A bought B"*. Strict entailment is the case when the conclusion can be inferred from the premises on the basis of logical deduction, possibly using world knowledge: *"A purchased B"* $\longrightarrow$ *"A owns B"*. Although the concept of "Textual Entailment" has been commonplace in logic-based QA for a long time, it has been recently adopted as the key evaluation criterion in the "Recognising Textual Entailment Challenge" [Dagan et al., 2005].[15]

[14] <http://ai-nlp.info.uniroma2.it/te/>

[15] <http://www.pascal-network.org/Challenges/RTE/>

1.3 Typical QA system

Most of the recently developed QA systems are organized around the following main modules [Hirschman and Gaizauskas, 2001a].

- Question Analysis and Classification

  The purpose of this module is to classify the question into one of the categories defined by the system, and transform it into a query suitable for Document Retrieval. At the same time, the expected type of the answer is determined (typically using a small set of answer types, e.g. person, organization, location, date, ...).

- Document Collection Preprocessing

  The document collection might need to be pre-processed to transform it into a form which is appropriate for real time QA. Various degrees of analysis are possible, from a simple indexing to a full parsing of the entire collection.

- Candidate Document Selection

  The purpose of this module is to extract from the document collection a limited set of documents that are likely to contain the answer. Traditional IR techniques can be employed, as well as advanced search engines.

- Candidate Document Analysis

  Within the retrieved documents, it might be necessary to locate sentences or passages that are directly related to the question, and contain entities of the type determined by the question classifier. These specific sentences or passages are then passed on to the next module.

- Answer Extraction

  The purpose of the Answer Extraction module is to locate within a small set of passages or sentences the units (words, phrases, text spans) that can be a direct answer to the question. Often this module makes use of advanced NLP techniques.

- Response Generation

  The extracted answers need to be presented to the user, possibly with some context from the source documents, or a generated explanation.

1.4 Overview of the dissertation

The process of transformation of textual documents into the representation of choice is based upon various techniques from the field of Natural Language Processing, none of which is at present capable of delivering perfect results, so that some amount of error in unavoidable. The aim of our work is to reduce the degree of error and at the same time offer to the end-users a system which can be employed in a real-world context, to increase their productivity by reducing the time required for finding relevant information. The research area of "Question Answering" (QA) tries to provide a partial solution to this problem by offering users a chance to find relevant information from a document collection using queries expressed in Natural Language. In particular since 1999 these activities have been encouraged by a specialized track within the US-funded TREC competition. Chapter 2, "Overview of Question Answering Systems" of the dissertation provides a detailed survey of such research efforts.

ExtrAns, described in detail in Section 2.8, can be considered as a specialized QA system, planned and in large part developed before the mainstream QA

research really took off. Many of the ideas and techniques further developed by the author were already present in the original ExtrAns system.

In particular, the entire concept of Answer Extraction and its implementation had been developed before the start of this dissertation. However, none of the ideas concerning domain descriptions and their role in technical texts, including the problems around them and the ideas for their disambiguation presented in this dissertation were present in the original ExtrAns system.

Although the techniques adopted in the original ExtrAns system were not necessarily tailored to a domain-restricted application, the choice of a specific domain is potentially an advantage, as it allows us to make use of specialized resources, in particular terminologies and ontologies, which in some cases are already available. The problem of semantically grounding the entities of discourse can be solved by making use of such resources. For example, all parts of an aircraft, although they might be named using different terms, will have a unique identifier, which can be taken as their referent, and can be traced within standard databases. In a similar fashion, many elements of scientific discourse, in particular in the biomedical field, are increasingly given official identifiers in standard resources. A good example is UniProt, a comprehensive catalog of information on proteins, where each protein is given a unique identifier.

However, the variability of technical descriptions poses challenges in its own right, which are not usually dealt with by open-domain QA systems. In Chapter 3, "Domain Specific Collocations and Technical Terminology" we introduce in more detail the notion of domain description, in relation to terminology and domain specific collocations. We also present a survey of previous work by other authors aimed at extracting domain specific relationships from sets of domain descriptions or from document collections.

Techniques for terminology extraction play an important role in the recogni-

tion of domain descriptions, even though they are far from being a complete solution to the problem of identification. In Chapter 4, "Extraction of Domain Descriptions", we present and evaluate two tools intended to extract terminology, which we have used within the scope of a practical application, as a support in the extraction of domain descriptions. Most of the so-called terminology extraction tools do not (and cannot) extract terminology, as there is no operational definition for what constitutes terminology: they simply propose a list of candidate terms (mainly multi word expressions culled from the documents representing the domain) and leave to the user the task of selection.

The domain descriptions obtained using terminology extraction tools and manual validation are of limited use without the capability of recognizing their interrelationship. While different (but often related) domain descriptions can be used to refer to the same domain entity, it is necessary to recognize all possible variants and possibly structure them in a way that they can be used in the question answering process. Specific types of relations, such as *subtype of* or *part of* can also prove to be useful.

In Chapter 5, "Structuring of domain descriptions", we present techniques that we used in order to recognize some of these important relations. Of course, the objects of the domain are interconnected by a large number of other domain relations, which, if available, would greatly increase the capabilities of a knowledge-based question answering system. However, within the scope of the present work we have limited ourselves to the simpler relations mentioned above.

Finally, the domain descriptions and their relationships can be put to good use within a query system. For example, the Aircraft Maintenance Manual (AMM) of the Airbus A320, which in source form is approximately 120MB large, describes how the constituent parts of the aircraft relate to each other,

the testing and maintenance procedures for each part, as well as the tools and materials to be used. As 30% of the words in the running text belong to the descriptions [Rinaldi et al., 2002a], pre-processing needs to be focused in this direction. Extraction of the domain descriptions, followed by thesaurus construction are necessary first steps before using the resources so obtained in the question answering process.

In Chapter 6, "Domain descriptions in the QA process", we present a question answering system which makes use of domain descriptions and relationships extracted from the AMM and allows a flexible natural-language based querying of the manual. The success of this application lead us to consider other corpora and other domains rich in domain descriptions.

The life sciences domain, with its wealth of terminological resources appeared particularly apt for a further demonstration of the role of domain descriptions in the question answering process. In Chapter 7, "The biomedical domain: motivations and background research" we survey a number of NLP-based approaches for targeted text mining over biomedical literature, including applications such as entity detection and relation mining. In Chapter 8, "A QA application for biomedical literature" we describe an application of the domain descriptions techniques and of our QA system to a biomedical corpus. That initial experience made us aware of the importance of domain descriptions not only in the QA task, but also for other applications. In Chapter 9, "Relation Mining over Biomedical Literature" we describe a relation mining application over biomedical literature where domain descriptions play a crucial role for the correct identification of more complex domain relations. We also briefly mention more recent research activities carried out by the author, which further extend the results presented in this dissertation. We conclude with a brief summary of the work described in this dissertation (Chapter 10, "Conclusion").

Chapter 2

Overview of Question Answering Systems

As mentioned in the introduction, the background to the work reported is an identification of some of the major problems with existing QA system, which was based on a survey conducted by the author and summarized here.

2.1 Early QA work

Pioneering work on QA was performed in the 1960s, in particular in the area of Natural Language Interfaces to Expert Systems and to Database Systems. Already [Simmons, 1965] in 1965 could survey 15 QA systems that had been implemented. Perhaps the best-known example of a such system is BASEBALL [Green, 1961], which could answer simple questions about baseball games played over one season of the American league. The information about the games was stored in a database. The system transformed the questions into a canonical form which was used as a query into the database.

Another influential early system was LUNAR [Woods, 1977], which was designed to answer questions about the geology of lunar rocks obtained from the Apollo moon missions. The background information was again stored in a database.

A parallel area of research was practiced with dialogue systems, which aim at answering questions as part of an interaction with the user, as opposed to one-off questions. An early example of this approach was SHRDLU [Winograd, 1972], which simulated a robot acting in a toy world of geometric objects. The user was allowed to interact with the robot through simple instructions. SHRDLU's grammar is based on the notion of systemic grammar, a system of choice networks that specifies the features of a syntactic unit, how the unit functions, and how it influences other units. SHRDLU demonstrated that a computer could carry on a simple conversation about a blocks world in written English.

Another early system, which used very simple techniques to emulate a conversational agent, was Eliza [Weizenbaum, 1966]. Eliza has been described as a 'parody' of a psychotherapist. The system, on the basis of a very simple parsing of the previous utterance, will formulate a question or statement, that appears to be a logical continuation of the dialogue (at least in some cases). The success of systems like Eliza and SHRDLU led to claims that NLP had been solved and predictions that within a short time conversations with computers would be just like those with people.

Another early dialogue system was GUS [Bobrow, 1977], which simulated a travel advisor, helping the user to locate a suitable flight. A description of a generic QA algorithm appears in [Simmons, 1973]. Modern concerns, such as question classification, also appear in early work, such as [Lehnert, 1978]. Her approach was based on Schank's framework of scripts and plans [Schank, 1975, Schank and Abelson, 1977]. She devised a theory of

question answering and an implementation of that theory in a system called QUALM [Lehnert, 1978]. Her question classification was later extended by [Kolodner, 1980] and [Dyer, 1983a]. The latter offers a QA system (BORIS) over short narratives. All early systems had very limited scope: typically they were applied to a *restricted domain* and allowing only *toy questions*.[1]

A general problem of all these systems (and indeed of all NLP-based system) is that in all but the most trivial cases, comprehension of a text, or a section thereof, requires knowledge of the objects and situations that are described (including abstract knowledge and knowledge about relations between elements in the world). This type of knowledge (often referred to as *world knowledge*) is a prerequisite not only for making complex inferences, but sometimes also for simple semantic disambiguation. Consider the well-known example: *"The soldiers shot the women. They fell down."*. This sentence has two possible interpretations (namely: *"The soldiers fell"*, or *"The women fell"*). The sentence can only be disambiguated on the basis of the fact that in general someone who is shot is more likely to fall than someone who shoots. In this case, the disambiguation of the sentence is based purely on contextual knowledge, and the conclusion can be considered only statistically correct (i.e. more likely than the other reading).

Another more general example is given by the sentence *"Open on Sundays"*, which can be found as a notice on the front door of a shop. The intended interpretation is, generally, that the shop is open on Sundays *additionally* to its regular opening days. But such an interpretation depends on a whole set of assumptions derived from our world knowledge.

As it is impossible to give to a computer access to a sufficiently large body

[1]The early systems typically allowed only a very limited range of question, with only minimal variations. These questions are occasionally referred to as 'toy questions'.

of world knowledge,[2] one way to help a system make use of limited world knowledge for NLP purposes is to 'package' items of knowledge that find wide applications into simple units which can then be represented as data structures.

In 1972 Roger Schank presented a theory of human language processing called "Conceptual Dependency Theory" (CD) [Schank, 1972], which is based on the assumption that natural language understanding is largely predictive (i.e. based on expectations) and that syntax is only used as a pointer to the conceptual meaning.[3] The purpose of the CD Theory is to offer a means to represent information at the conceptual level, based on a limited number of primitives. Two major categories are considered: picture producers (objects, typically represented by nouns) and actions (typically represented by verbs), which are connected by a limited set of *conceptual dependencies*.

In [Schank and Abelson, 1977], the authors make use of knowledge structures that they call scripts that represent stereotypical situations. For example the knowledge that a telescope is normally an instrument used for seeing remote objects could be encoded in such a script. Scripts connect a sequence of actions related to a well-known situations (e.g. going to a restaurant) and they allow us (or a machine) to understand a situation without requiring complex inferences.[4] The existence of these structures allows conceptual inferences to take place. This type of inference [Rieger, 1975] is not

[2]Recent developments could in part challenge this statement, as computers can nowadays access a very large body of unstructured knowledge, in the form of the World Wide Web. However, whether they can make use of it in a meaningful way, is still an open research question.

[3]Notice that this assumption is in stark contrast with most work in linguistic theory (and consequently large part of NLP), which presumes that syntax can be treated autonomously from semantics.

[4]For example, the interpretation of the word *"check"* can be different according to whether the customer in a restaurant asks for it, or offers to pay with it.

necessarily a logically valid deduction, and might even lead to contradictions. Its main purpose is to fill in information missing from the utterance, based on the information explicitly provided and the background information. The truth value of every inference cannot simply be classified as true or false, but rather has a degree of likelihood, based on the premises from which it is drawn.

The BORIS system, presented by Dyer in [Dyer, 1983b], deals with legal questions. The system is proposed as an implementation of a cognitive theory developed by the author. It is based on knowledge structures called 'Thematic Abstraction Units' [Dyer, 1983c], and tries to model different aspects of human cognition, such as plans, interpersonal relationships, emotional reactions, social roles, etc. Like similar systems of the same age, it makes heavy use of hand-coded knowledge, in the form of scripts, goals, settings, etc. Another well-known variation of the same approach is that presented by Wendy Lehnert [Lehnert, 1982] as 'Plot Units'.

The problem that all these systems try to solve is that only a fraction of the knowledge needed to understand a story is provided explicitly within the story itself. It is therefore necessary to provide the system with additional knowledge, so that inferences can be drawn on the basis of the known facts, and the information provided in the input text. The script (or thematic abstraction unit, or plot unit) is intended as a support for inference generation. All of these approaches lack a description of how these structures can be acquired or constructed in a more general fashion: it is very clear that the choice of what elements to include in the knowledge structures and of what internal structure to impose on these elements is a matter of idiosyncratic choice by the designers of the system. Furthermore, the problem of how to move beyond the limited domains of the implemented prototypes is not considered.

An additional problem of some of the early systems is that the problems involved in producing the internal representation, starting from natural language input, were overlooked. Often it was simply taken for granted that a simple transformation was possible, and this allowed the researchers to focus on the problems of knowledge representation and reasoning, ignoring the crucial problems of language understanding.

For example, the system developed by E. Charniak for his doctoral dissertation [Charniak, 1972] could handle simple children's stories, but it did not accept sentences in English as input, instead it used a format similar to the internal representation. Similar shortcuts are typical of most of the systems developed in the 70s and early 80s. The developers were driven by an amount of optimism that some of the problems that they ignored would quickly be solved, so that they could focus on those aspects that were most of interest to them.

Hubert Dreyfus, an early skeptic of the AI efforts [Dreyfus, 1992], wrote: "[...] the formulation of a theory of common sense was a much harder problem than expected. It was not, as Minsky had hoped [Minsky, 1975], just a question of cataloging a few hundred thousand facts. With years of hindsight and experience, we now understand better why the early optimism was unrealistic. Language, like many human capabilities, is far more intricate and subtle than it appears on first inspection [Winograd, 1987]."

2.2 Developments in the 80s

In the late 80s a few systems continued this line of knowledge-heavy inference-based approaches. Amongst the best known examples are: Faustus [Norvig, 1987], the Unix Consultant [Wilensky et al., 1984,Wilensky et al., 1988], TACITUS [Hobbs et al., 1988], MIDAS [Martin, 1990]. During this decade there

was a gradual shift from rigid hard-coded knowledge to the adoption of weighted or probability-based approaches.

Faustus [Norvig, 1987] (Frame Activated Unified STory Understanding System) attempts to unify different knowledge structures under a common frame-based representation format. The input sentences are parsed using the PHRAN natural language analyzer and the results of the analysis is mapped onto existing frames. For example, a story containing a sentence like *"Ann went to watch a movie"*, is mapped onto a 'movie-going' frame. Different frames are activated by the input sentences, and this activation is signaled by 'markers'. Related frames are then selected by a process of spreading activation, so that as new sentences are provided, the internal representation of the state of affairs provided by the input is modified. This approach is based on a spreading activation mechanism, which can result in collisions, because multiple interpretations of the input are possible. When collisions occur, an inference can be drawn, which rules out some of the possible interpretations. At the end of this process, the network is left in a coherent state, which correspond in general to the desired interpretation of the input. If multiple interpretations are possible, a measure of 'coherence' is used to select a set of inferences which maximizes this value. The spreading activation model was inspired by the TLC program of Quillian [Quillian, 1968].

The Unix Consultant (UC) [Wilensky et al., 1984, Wilensky et al., 1988] was a project aimed at offering an interactive support system for naive UNIX users. The users can ask plain English question, to which the systems responds by generating an answer in English. The user's input is parsed using the ALANA language analyzer, which produces a semantic representation of the question in the form of a KODIAK semantic network (a knowledge representation formalism developed at Berkeley by R. Wilenksy [Wilensky,

1987]). Reasoning is based on a component called a 'concretion mechanism' [Wilensky, 1983, Norvig, 1983], which is capable of further refining the content of the input query. Internally, the system maintains a model of the user, based on a goal analyzer (PAGAN), and a planner (UCEgo), which creates strategies for satisfying the system's goals. A language generation component (UCGen) is capable of verbalizing a response to the user's query from the internal knowledge structures and the plan generated in order to satisfy his knowledge request.

TACITUS [Hobbs et al., 1988, Hobbs et al., 1993] makes use of *abductive inferences*, trying to find the best explanation for a given possible conclusion. It uses weighted logical forms, where each logical construct is given a cost: the higher the cost, the more unlikely a given assumption. Interpretation of the logical form consists in determining values for the logical variables that minimize the total cost. The system makes use of a large, comprehensive lexicon (including subcategorization information) and a comprehensive grammar of English. It produces a parse and semantic interpretation of each sentence in which different possible predicate-argument relations are associated with different syntactic structures.

[Charniak and Goldman, 1989] proposes a similar model, replacing the 'costs' with standard probabilities (which have a sounder mathematical foundation). The sentence interpretation is based on Bayesian probability theory. They apply this approach to the construction of belief networks (which amount to probability distributions represented as directed acyclic graphs).

A rather more recent example of a system partially based on the QA paradigm is FAQFinder [Burke et al., 1997], which matches user questions against given question/answer pairs (taken from the 'Frequently Asked Questions' posts of newsgroups). The approach is based on finding the most similar question, and then providing the corresponding answer. Therefore, al-

though superficially similar to current QA work, the approach is substantially different, in that it is limited to find similarities between questions, and does not deal with the problem of selecting the best answer.

2.3 Question Answering in TREC

Since 1992 the U.S. Department of Defense and the National Institute of Standards and Technology (NIST) have been sponsoring the annual Text REtrieval Conference (TREC) as part of the TIPSTER program. The TREC conferences are organized as competitions among participating systems to solve a common problem, established by the organizers, who also provide the basic infrastructure, in terms of document collections and standard evaluation procedures. The main aims of TREC are: to foster research within the information retrieval (IR) community, to increase communication among industry and academia, and to promote technology transfer.

Each TREC conference consists of several *tracks*, each of which focuses on a particular aspect of the retrieval problem. Examples of tracks are the SPAM track, meant to evaluate SPAM filtering techniques, or the Genomics track, focusing on IR over biomedical documentation. Over the course of TREC some tracks have been discontinued and new tracks have been started. A new track serves as an incubator for novel research areas, defining the scope of the problem, and creating the necessary infrastructure.

Of particular interest to us is the Question Answering track [Voorhees, 2001] which has been running since TREC-8 (1999). Broadly speaking, the aim of this track is to compare systems on the task of retrieving *answers* from document collections, rather than retrieving documents in which the user has to find the answer. While conventional IR systems take as input a description of the "Information Need" of the user (normally provided as a lengthy de-

scription), and retrieve relevant documents from a document collection, this track restricts the description of the information need to a single question, and the answer to a small text snippet.

Compared to the early QA systems which we described in Section 2.1, the main conceptual difference is the replacement of the knowledge base with a document collection as source of the background knowledge. Although in part due to the origin of the QA track within an information retrieval challenge, this is also a major step forward in terms of what can be expected from information retrieval systems in general. It greatly increases the flexibility of the system and removes the need to manually encode the knowledge upon which the answers are based.

Even if during the course of TREC a few different subtasks have been considered, the main task has remained in large part stable. The participating systems are requested to deliver 5 ranked answers for each question, together with the identifier of a document within the collection supporting the answer provided. The answer itself has been defined as a text snippet containing the expected answer, although the precise definition of such a text snippet has evolved since the first TREC.

The organizers provide the document collection, mainly based on newspaper articles, the list of questions (derived from web logs of search engines or other sources), and guarantee a fair evaluation. Additionally, they provide the ranking of the documents in the collection (the first 1000, except in TREC-8, when it was the first 200) obtained using the question as a query in an independent information retrieval engine. In TREC-8 and 9 the SMART system was used, from TREC 10 onwards the PRISE system was used. This information is meant as a service for the groups which do not have the possibility of using an IR engine of their choice.

The evaluation of the results makes use of the measure known as Mean

Reciprocal Rank (MRR). As previously mentioned, for each question the participants must return 5 ranked answers. For each question the score of the answer is the reciprocal (1/N) of the rank N (position from 1 to 5) at which the first correct answer is found, or 0 if no correct answer is found. The MRR measure is defined as the mean of the scores over all the questions. Additionally the number of questions where no correct answer was found is also reported.

In the following sections we briefly discuss the individual differences across each of the TREC-QA competitions. It is remarkable however that there appear to be no major conceptual differences in the systems presented in recent competitions compared to those presented in the earlier competitions. Systems tend to improve on earlier ideas, without any radical modification, other than what is needed to cope with the variations in the definition of the task.

2.3.1 TREC 8, 1999

The first edition of the TREC QA track was organized as part of TREC 8 in 1999.[5] The organizers selected 200 questions from a total of 1500 questions proposed by NIST and the participants, considering also as a source the logs from the FAQFinder system from Cornell University. There was no thorough control of the questions against the document collection, so that later a few of them were found to have no answer (or ambiguous answers). Those were excluded from the evaluation.

The answers were defined as text snippets of fixed length, either 250 bytes or 50 bytes, which were evaluated separately. In other words, precise answers were not expected, rather a "text window" containing the answer had to

[5] <http://trec.nist.gov/pubs/trec8/t8_proceedings.html>

be provided. In cases where multiple entities of the same semantic type as that of the expected answer were contained in the snippet, with no means of deciding which was the preferred one, the answer was considered wrong. Thus, systems did not get credit for multiple correct answers, or for being able to identify that no answer could be found in the collection. The evaluation was carried out by human assessors, each answer being evaluated by three assessors. Some degree of disagreement is unavoidably associated with this process.

The best results were obtained by the system presented by [Srihari and Li, 2000] in the 250 bytes track, and by [Moldovan et al., 2000] in the 50 bytes track. Most of the successful systems adopted a strategy centered on the identification of the type of answer expected by the question (person, location, etc.), and the location in the candidate passages of entities belonging to that type. Interestingly, systems which attempted some form of deep parsing obtained comparatively poor results. For example, the Sheffield system [Humphreys et al., 2001] reached a mean reciprocal rank of 0.111.

2.3.2 TREC 9, 2000

In TREC 9[6] the number of questions was increased to 693 and the background collection was extended to include documents from the Foreign Broadcast Information Service, Los Angeles Times, Financial Times, Wall Street Journal, Associated Press Newswire and San Jose Mercury News, totalling about 1GB of compressed data). The questions were partly drawn from an Encarta log, made available by Microsoft and partly created using a log of the search engine Excite as a source of ideas. All questions were checked against the document collection to assure that they could be an-

[6] <http://trec.nist.gov/pubs/trec9/t9_proceedings.html>

swered.

It was also decided to test if variations in a question would influence the capability of systems to answer that question. For this, 54 of the original 500 questions derived from the logs were reformulated to generate 193 variants, which were added to the set of questions. For example, the test set contained the following variants of the same question [Voorhees, 2001]: *"What is the tallest mountain?"*, *"What is the world's highest peak?"*, *"What is the highest mountain in the world?"*, *"Name the highest mountain."*, *"What is the name of the tallest mountain in the world?"*. Systems capable of detecting the similarity of questions could therefore profit from such information.

The answers were still requested as text snippets as in TREC 8 (either 50 or 250 bytes), but the judgments were tightened because the rules of TREC 8 did not clearly specify how to consider an answer that was correct but could not possibly have been derived from the document provided as evidence. Because of this unclear specification, such answers were considered correct in TREC 8, while in TREC 9 they were considered as unsupported. Systems were then compared using two separate criteria: *strict*, where only correct and supported answers were evaluated positively, and *lenient*, i.e. where correct but unsupported answers were also evaluated positively.

The system that achieved the best results is described in [Harabagiu et al., 2000]; it is based upon a combination of NLP and Knowledge Engineering techniques (for a more detailed description see Section 2.4).

2.3.3 TREC 10, 2001

The 2001 edition[7] saw as the main innovation the dropping of the 250-bytes window task, and the loosening of the length requirement for the 50-bytes

[7] <http://trec.nist.gov/pubs/trec10/t10_proceedings.html>

window task. In the case of the 250 bytes window, the experience of the previous two editions had showed that the task was not significantly different from a traditional IR task. Simple passage-based IR consistently outperformed other techniques. In the case of the smaller window, the 50-bytes length was now intended as a maximal length of the answer, rather than as a fixed length. This change allowed systems to be more precise in pinpointing the answer. Another innovation was the fact that credit was given for being able to identify that no answer could be found in the collection. Systems were allowed to return the value of NIL to indicate this option. This value was marked as correct if there was no known answer in the collection, and wrong otherwise.

Additionally, two subtasks were introduced, with a slightly different focus. The *context question task* provided sets of related and sequential questions. The *list task* expected as answers not a single item but rather a list of suitable items (instances of a particular kind). It requires systems to assemble an answer from information located in multiple documents.

Contrary to the expectations of the organizers, very few participants tackled the additional subtasks and most of them used the same systems as for the main task, so that no new insights could be won. Only in TREC13 the introduction of 'context' for questions, in a different format, turned out to be successful.

Among the results, the performance of [Subbotin and Subbotin, 2001] was remarkable, obtaining the best results (70.1%, lenient score). The system is based on pattern matching techniques derived from text summarisation. For example, in order to detect an answer to the question *"When was X born?"*, a pattern such as *"X (AAAA-BBBB)"* was used (where AAAA would be the year of birth).

2.3.4 TREC 11, 2002

In 2002 the QA track[8] moved further towards the goal of identifying precise answers, by requiring systems to return a single noun or noun phrase, rather than a text snippet. Given this stricter requirement, extremely precise named entity recognition was essential to achieve good results. The best performing system was [Moldovan et al., 2002], which was able to find 83% correct answers, while the next best system [Yang and Chua, 2003] achieved only 58%. The document collection used as the source of answers was the AQUAINT Corpus of English News Text [Graff, 2002]. "This collection consists of documents from three different sources: the AP newswire (1998-2000), the New York Times newswire (1998-2000), and the English portion of the Xinhua News Agency (1996-2000). There are approximately 1,033,000 documents and 3 gigabytes of text in the collection. The test set of questions contained 413 questions drawn from AOL and MSN Search logs." [Voorhees, 2003]

2.3.5 TREC 12, 2003

In the next edition of TREC (2003),[9] it was recognised that the task of pinpointing the right answer, and the problem of finding a passage which contain the right answer, could both provide interesting applications, while perhaps requiring different technologies. Therefore the track was divided into two tasks: the passages task and the main task. The document collection used as the source of answers was again the AQUAINT Corpus of English News Text. The same set of questions was used for both tasks.

In the passages task, as in the first two editions of the QA track, systems

[8] <http://trec.nist.gov/pubs/trec11/t11_proceedings.html>

[9] <http://trec.nist.gov/pubs/trec12/t12_proceedings.html>

returned a single text snippet of 250 bytes in response to factoid questions. The only difference was that rather than requiring it as a text string, systems were required to provide a document identifier and an offset within that document. This avoided the problem of checking whether the answer did actually come from one of the documents of the collection. The evaluation metric was the number of snippets that contained a correct answer.

The questions in the main task were organized into three categories: *factoid* questions, *list* questions, and *definition* questions. The different types of questions were evaluated separately, and an overall score was computed combining the scores of each of the types [Voorhees, 2003]. This was done in order to encourage participation in the additional subtasks. For example, the list task had been running since 2001, but with very sparse participation. Unlike the previous two editions, systems were allowed to return an arbitrary number of answers to list questions.

TREC 12 introduced for the first time 'definition questions', such as *"Who is Colin Powell?"* or *"What is mold?"* [Voorhees, 2003]. It had been previously observed that definition questions are one of the most common question types which can be found in search engines logs. The problem in using definition questions in a formal evaluation is that the answers are very difficult to evaluate. A simple 'right' or 'wrong' judgment is no longer appropriate [Voorhees, 2003].

2.3.6 TREC 13, 2004

In 2004 the QA track was partially modified with respect to the previous editions[10]. Questions were combined into question series, which had a given target associated with them. A target belongs to one of the possible answer

[10] <http://trec.nist.gov/pubs/trec13/t13_proceedings.html>

types (`person`, `organization`, `thing`).

A question series aims at finding information about its target using questions of different type (factoid and list questions), each of which defines a possible 'facet' of the desired information. There is a final 'open' question which allow the participating systems to provide any additional information they know about the target, which has not yet been provided as answer to any of the previous questions (in part similar therefore to the 'definition' questions of the previous edition). A separate evaluation for each question type was performed, and a global score was computed as the weighted average of the different component scores.

It is interesting to notice that organizing the questions into series provides a new interesting perspective over the whole task. In fact, each series can be seen as an abstraction of a dialog, where a user tries to elicit additional information about a given subject. The previous questions and their answers provide the context for the current questions [Voorhees, 2004].

Once again, the AQUAINT corpus was used as the background document collection.

2.3.7 TREC 14, 2005

The same approach adopted in 2004 was repeated in the main task of 2005,[11] with questions series which combined different types of questions. The only significant difference was that events (intended as a complex state of affairs, not as a single atomic event) were also included among the targets. Additionally, two independent subtasks were introduced: *Document Ranking* and *Relationship*.

The Document Ranking Task had the goal of creating pools of documents

[11] <http://trec.nist.gov/pubs/trec14/t14_proceedings.html>

containing answers to questions in the main series, to be used for later evaluation challenges. It was in this sense a traditional IR task, with the questions used as 'topics' for the participant systems. The assessors pooled together all the documents returned by the participants, manually evaluated them for their relevance to the query, and scored the systems on the basis of the resulting pools.

The Relationship task aimed at identifying relevant connections between entities. A Relationship was defined as the ability of one entity to influence another, including both the means to influence and the motivation for doing so. A limited number of relation templates was initially designed, and later filled with specific instances. An example of a template could be *"What evidence is there for transport of [goods] from [entity] to [entity]?"*, whereas a specific instantiation would be *"What evidence is there for transport of drugs from Mexico to the U.S.?"*

2.3.8 TREC 15, 2006

The structure of the main task was not changed substantially in 2006.[12] The main innovation was that temporal awareness was for the first time introduced in the evaluation criteria. Up to this point, an answer was judged correct if supported by a single document in the collection. However, in 2006 the criteria for correctness was that an answer had to be correct with respect to the most recent document in the collection.

For example, the query *"Who is the president of the United States?"* could have different answers in the documents (*"Ronald Reagan"*, *"George Bush"*, *"Bill Clinton"*, etc.) according to the date of the document. All of them were previously considered correct. In the 2008 evaluation, only the president

[12] <http://trec.nist.gov/pubs/trec15/t15_proceedings.html>

mentioned in the most recent document in the collection was considered a correct answer.

A new 'interactive' task was introduced, which allowed participants to refine their results through a single interaction with the assessors. The evaluation was performed in two stages. First participants submitted their initial runs and interaction forms to be filled by the assessors. The results of the interaction were returned to the participants, who could use them to improve their results and submit a second run. Both runs were then evaluated by NIST. New evaluation metrics were adopted for this task: modified F-scores based on nugget pyramids and recall plots based on response length (for a detailed description see [Dang et al., 2007]).

It is perhaps interesting to notice that system scores of the participants to the main task have been declining since 2004, rather than improving. This has been attributed by the organizers to the increasing difficulty of the task, and greater strictness of the evaluation process.

2.3.9 TREC 16 and beyond

The proceedings of TREC 16 (2007) are not yet available, so it is impossible at this stage to delineate a precise picture of the QA track. On the basis of the conclusions of TREC 15, and some participant's reports available on the web, it can be concluded that in 2007 some significant changes were introduced. The 'interaction' task introduced in 2006 was probably extended and made more dynamic. Rather than a 'frozen' interaction at NIST's headquarters a 'live' interaction through the participants web sites was attempted. The main task was also overhauled, not in its structure, but rather in the genre of documents. A move towards newswires and blogs was proposed, in order to introduce a new level of challenge for the partici-

pants, who would have to deal with informal, irregular language.

Although the motivations are not yet completely clear, it is important to notice that a decision has been taken to discontinue the QA track altogether. A new 'blog' track has been introduced in its place in TREC 2008.

The QA tracks had undoubtedly the benefit of promoting advanced research in an innovative area. The constant introduction of new tasks of increasing complexity guaranteed that the participants had to come up with novel solutions each time.

However, a possible criticism of all such large scale evaluation challenges is that, by attempting to provide a realistic scenario, it becomes increasingly complex to filter out what it is exactly that they are evaluating. In other words, since the overall results of the participating systems depend on a myriad of different components, it is difficult to draw conclusions as to what specific technologies add most to the results, which of them are rife for widespread usage, and which of them warrant further research investment.

2.3.10 The Standard Architecture

Most TREC systems are based on a common underlying conception, with a few variations among them, so that it is possible to speak of a sort of Standard Architecture for text-based QA systems [Abney et al., 2000, Pasca and Harabagiu, 2001]. Already after the second edition, some sort of convergence seemed to emerge, towards such a common base architecture, centered on four core components [Voorhees, 2001, Hirschman and Gaizauskas, 2001b].

A *Passage Retrieval* [Clarke et al., 2001a] module is used to identify paragraphs (or text windows) that show similarity to the question (according to some system specific metric), a *Question Classification* module [Hermjakob,

2001] is used to detect possible answer types, an *Entity Extraction* [Kim et al., 2001] module analyses the passages and extracts all the entities that are potential answers and finally a *Scoring module* [Breck et al., 2001] ranks these entities against the question type, thus leading to the selection of the answer(s).

Systems vary in the relative importance of each of the modules. The two possible extremes are plain IR and full NLP. In the former case, documents are broken into small units, which are then matched against the query. The problem is how to define a unit that is small enough to be considered an answer and yet large enough that it can be indexed efficiently. In the latter case, all documents would be fully analyzed, transformed into a semantic representation, and the query would have to be satisfied against this semantic representation.

It is interesting to observe that no participant ever considered a linguistic processing of the entire document collection: its sheer size makes a complete analysis very time consuming. Besides, a generic document representation that would allow an efficient storage and retrieval of the knowledge items needed to answer a given question is not yet available.

The usage of information retrieval techniques to obtain a relevant set of documents (or passages) for further processing was therefore a necessity. All participants which made use of linguistic techniques adopted conventional IR as a pre-filter to obtain a manageable set of documents.

On the other hand, another interesting observation is that the identification of an exact answer could not be obtained with information retrieval techniques alone. Systems that applied linguistic processing to the selected document fragments and questions performed consistently better than systems that relied on plain IR.

However, the most successful systems did not in general use a traditional

approach based on deep syntactic parsing and some form of semantic representation, but rather used an array of NLP techniques which differently contributed to their results. Among the most frequently used techniques we can identify the following. *Query expansion* (e.g. based on synonyms found in WordNet) was used to improve the results of passage selection. *Shallow parse* of the question (and candidate document fragments) was used in order to obtain some form of structural matching. *Question classification* was used to identify the type of entities required by the question. *Named-entity recognition* played a crucial role in order to recognize the type of entities in the candidate passages and match them against the required question type.

2.3.11 Lessons learnt from the TREC QA track

IR techniques can be used to implement QA/AE systems, by applying them at the passage or sentence level. Portions of text with the maximum overlap of question terms contain, with a certain probability, an answer.

The TREC Question Answering Track demonstrated from an early stage the weakness of traditional IR approaches when applied to extracting answers from documents [Voorhees, 2000b, Voorhees and Harman, 2001]. This inadequacy of IR techniques is most visible when answers have to be found within a small window of text (50 bytes). It turned out that systems that used some form of deeper linguistic knowledge can do a good job when the answer had to be localised within a small snippet of text.

The standard methods used in IR to rank hits according to their relevance are no substitute for these techniques. Relevance in IR is almost invariably determined on the basis of the weights assigned to individual terms, and these weights are computed from term frequencies in the documents (or passages) and in the entire document collection (the tf/idf measure). Since

this measure is blind to syntactic (and hence semantic) relationships it does not distinguish between hits that are logically correct and others that are purely coincidental. 'Bag of words' approaches will never be able to distinguish different strings that contain the same words in different syntactic configurations, such as 'absence of evidence' and 'evidence of absence'.

Thus, the experience of TREC tends to confirm the utility of NLP in finding answers, the problem is that the NLP systems are not yet fast and robust enough to cope with the volume of documents considered by TREC. Furthermore, problems such as synonymy, paraphrasing and inference require knowledge that goes beyond pure NLP.

It is interesting to observe how some of the systems that obtained good results in the QA track of TREC have gradually moved away from bag-of-words approaches and into NLP techniques, such as semantics and logical forms. For instance, Falcon [Harabagiu et al., 2001] (the best performing system in TREC 9) performs a complete analysis of a set of selected texts for each query and of the query itself and creates, after several intermediate steps, a logical representation inspired by the notation proposed by Hobbs. The syntax analysis in Falcon is based on a statistical parser [Collins, 1996], which delivers a dependency-based representation of the question, upon which a semantic representation is built.

As for the type of inferencing, Falcon uses an abductive backchaining mechanism, which can be used to provide a "logical proof" as a justification for the answer. Further, it also has an interesting module capable of caching answers and detecting question similarity. In an environment where the same question (in different formulations) is likely to be repeated a number of times such a module can significantly improve the (perceived) performance of a QA system.

Another idea that some QA systems have adopted is to use the web as a

resource. Several contributions to the QA track of TREC used the Web as a means to exploit data redundancy and avoid the need for complex linguistic analysis [Clarke et al., 2001b, Brill et al., 2001, Dumais et al., 2002, Lin, 2002]. The rationale is that, provided there is enough data, there will always be some passage that explicitly shows the answer to the question using a simple pattern. The Web becomes a knowledge resource that can be accessed by crawlers and search engines, and used for question answering.

An interesting observation is that for the vast majority of questions there isn't just one obviously correct answer. Many different answers are possible, at different levels of granularity. Whether any of them will satisfy the question depends to a large extent on the person receiving the answer. [Voorhees and Tice, 1999] refer to this problem as the *myth of the obvious answer*.

We now continue our survey of question answering research by presenting some of the most relevant QA systems and discussing the technologies that they adopt. We then conclude with a description of a restricted domain QA system (ExtrAns) which was used by the author for the experiments described later in this dissertation.

2.4 University of Texas / LCC

The system developed at the University of Texas [Moldovan et al., 2000], [Harabagiu et al., 2000] was undoubtedly one of the most interesting and most representative of the whole TREC-QA effort. It obtained the best results in TREC-8 and TREC-9, the second best results in TREC-10. Later a further development of the same system was presented by the same authors as a product of a company called 'Language Computer Corporation' (LCC). It was again the best system in TREC-11.

Keywords are extracted from the question in order to build a query for a

Boolean search engine. The search engine is customized in order to deliver candidate paragraphs, rather than documents. Initially the query is formulated quite broadly, but if the number of paragraphs returned is too large, the query is further refined by adding additional keywords until an ideal number of paragraphs is reached.

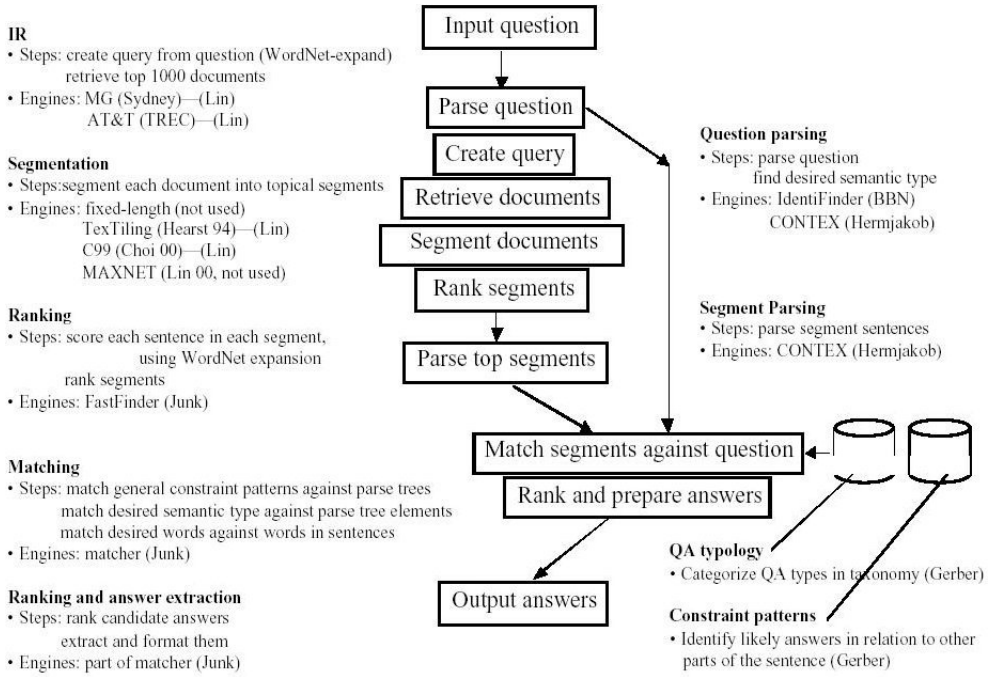
The retrieved paragraphs are then parsed and transformed into a logical representation. A coreference resolution component is used for detection of coreferential named entities and pronominal anaphora. The logical forms are derived from the parse trees and are close to the syntactic representation (e.g. they include subjects, objects, adjectival modification, prepositional attachments, etc.). Additional information is added using 'lexical chains', starting from the content words in the initial logical form. Lexical chains are built using WordNet relations and WordNet glosses.

The same procedure is applied to the question. Unification is then attempted between the logical form of the question and logical forms derived from the candidate paragraphs. In case of failure, a new query is built by expanding the keywords with morphological and lexical variations (such as synonyms derived from WordNet). The analysis process is then repeated with the results of the expanded query.

When candidate answers are found, a justification for them is built by abductive backchaining. The Otter logical prover is used to build a 'logical proof' of the answer, by finding the premises from which the answer was derived. When a complete justification is found, the corresponding candidate answer is considered to be the requested answer.

In TREC9 the system obtained comparatively high results. In the 50-byte task, the number of correct answers in the top 5 returned text snippets was 58% (strict evaluation) and 59.9% (lenient evaluation). In the 250-byte task, the results were 77.8% (strict) and 76% (lenient). The MRR was almost 0.6

Figure 2.1 . Architecture of Webclopedia (reproduced from [Hovy et al., 2000])



in the 50-byte task, and over 0.7 in the 250 byte task.

A special module capable of detecting question similarity might have played a major contribution to these results, as in TREC9 a significant percentage of questions were reformulations of other questions (see Section 2.3.2).

2.5 Webclopedia / ISI

The research group coordinated by Eduard Hovy at the Information Sciences Institute (ISI, University of Southern California) participated in TREC-9 with their system called Webclopedia [Hovy et al., 2000, Hovy et al., 2001]. The question is parsed using the CONTEX [Hermjakob and Mooney, 1997] sentence parser, which delivers a parse tree containing information about the part-of-speech of each word, their syntactic categories, and constituent

roles. CONTEX is additionally capable of assigning a type to the question, which can be used to detect the semantic type of the expected answer. For example, the question *"Who is the president of Switzerland?"* will be parsed by CONTEX producing the normal syntactic information and additionally the semantic type 'PROPERPERSON', which forces the candidate answers to contain names of individuals. A named entity extraction tool (BBN's IdentiFinder [Bikel et al., 1999]) is used to extract and classify all names in the parsed document (and in the question), using categories such as person, organization, role and location. For example, *"president"* will be tagged as 'role' and *"Switzerland"* as 'location'. At this point an IR query is generated, using keywords from the parsed question, which are expanded with synonyms using WordNet. For example, 'president' is expanded to

```
(president)|(head & of & state)|premier|leader
```

The query is submitted to the information retrieval engine MG [Witten et al., 1999], which is used to search the background corpus (the TREC-9 collection). If the number of documents returned by the query is above a specified threshold the original query is used. If nothing is returned, the query is relaxed and expanded in the hope of finding relevant documents.

The retrieved documents are segmented using TextTiling [Hearst, 1994] with the aim of splitting them into topical segments (semantically coherent units). Segments are then ranked where scores are assigned if a word in the segment matches with a word in the search engine query. Different types of words get different scores:

- each word in a segment that matches a word in the question get a score of 2;
- each word in a segment that is synonymous with a word in the question (but it is not identical with it) gets a score of 1;

- other words get a score of 0.

The top hundred segments, according to their ranking, are selected for further processing. CONTEX is again used to parse the selected segments, while BBN's IdentiFinder marks proper names (persons, organizations, locations).

Webclopedia makes use of a very refined QA Typology [Hovy et al., 2002, Hermjakob, 2002] consisting of hierarchically arranged QA types. Each QA type is associated with patterns of expression for both questions and answers in the form of templates. Other systems make use a taxonomy of question types [Srihari and Li, 2000, Moldovan et al., 2000], however the authors of Webclopedia claim that their system is far more sophisticated in this respect. Their taxonomy, rather than being based simply on the question word, attempts to represent the user intentions, with classes such Why-Famous (for the question *"Who was Christopher Columbus?"*, but not for *"Who discovered America?"*). The analysis of the questions delivers for each of them the associated answer templates (QA patterns) and a ranked list of Qtargets (the types in the QA taxonomy). Some Qtargets have additional roles associated with them (Qargs).

Webclopedia uses three different approaches to the final stage of matching question and candidate answers. The first approach makes use of the QA patterns, checking if there is a direct match in the candidate answers. The second approach exploits the Qtargets and Qargs, to check if any of the entities in the candidate answers fulfill the requirements. The third approach is a sort of fallback, based on a word-level window which is moved across the unparsed text and computing a score at each position, based on various information, including the presence of question words, or their wordnet expansions.

In [Hovy et al., 2000] the authors report that writing QA patterns manually

is one of the major limitations of their system (the process is slow and tedious, and the results are not robust). In [Ravichandran and Hovy, 2002] they experiment with techniques to automatically learn such patterns from web pages containing both the question words and the answer.

2.6 START

START [Lin, 2002, Katz et al., 2003, Katz et al., 2005b, Katz et al., 2005a] is a Question Answering system that was developed prior to the beginning of the TREC QA track, and has later been adapted and participated in all editions since 2002. Since the initial system aimed at finding an answer to a question from either its internal knowledge base or the web, it had to be extended in order to support a phase of so-called *answer projection*, which basically amounts to finding in the background collection answers which are equivalent to those found by the system. An online demo is available at <http://www.ai.mit.edu/projects/infolab/ailab>.

START uses as internal knowledge representation a structure called "T expression", which represents a relationship among two entities in the format: `(subject relation object)`. It is also possible to use other T-expressions as subject or object. This format is equivalent to a binary predicate in a classical logical form, and is similar to the ternary expressions used in many knowledge representation approaches (including, most recently, RDF). Using T-expressions as subject or object can be seen as equivalent to nested logical forms. Although typically the relation corresponds to a verb in the original sentence, it is also possible to have different types of relations governed by prepositions or other special words, representing cases like possessive nouns, prepositional phrases, etc. [Katz, 1997].

In common with many types of shallow semantic representation, different

T-expressions can result from sentences which have the same meaning but different surface structure. For example the sentences *"A presents a gift to B"* and *"A presents B with a gift"* will result in the T-expressions shown in Example 2.6.1.

Example 2.6.1

```
((A present gift) to B)
((A present B) with gift)
```

In order to be able to treat such expressions as equivalent, transformation rules are required, which make explicit the relationships between alternate realizations of the arguments of verbs in the form of IFELSE statements. In START, such transformation rules are called S-rules and are stored in a component called 'rule base'. S-rules can function in either 'forward' or 'backward' mode. For example a rule like the one shown in Example 2.6.2 allows the T-expressions shown in Example 2.6.1 to be considered as equivalent by the system.

Example 2.6.2

```
IF ((subj present obj1) with obj2) THEN ((subj present
obj2) to obj1)
```

Additionally, it is possible to make use of 'semantic classes' in order to generalize the rules. For example, the verbs *"reward"*, *"give"* *"offer"* *"present"*, etc. belong to the semantic class `act_of_giving`, which can be used in an S-rule, as shown in Example 2.6.3.

Example 2.6.3

```
IF ((subj verb obj1) with obj2) THEN ((subj verb obj2)
to obj1) PROVIDED verb act_of_giving class
```

The information necessary for the construction of the T-expressions is obtained by parsing the input documents. The parser used by the START sys-

tem is capable of breaking up sentences into smaller units, typically containing one single verb, called 'kernel sentences'. Over these smaller units, the system is capable of locating the subject, verb and object which are then used to build a single T-expression. Certain other grammatical categories like adjectives, possessive nouns and prepositional phrases are used to create additional T-expressions in which prepositions and several special words will serve as relations.

In the phase of question analysis, a similar approach is applied to the question. Additionally, a rule which corresponds to the reversal of the syntactic wh-movement (which defines the syntactic structure of wh-questions in English), is applied. First, the question words (*"who , whom and where"*) are located in the question, then they are relocated to the position where they would belong if the sentence was declarative, rather than interrogative. For example, the sentence *"Whom did John present with a gift?"* will be transformed into *"John present whom with a gift"*. Once the question has been transformed into a declarative sentence, T-expression generation is applied on the declarative sentence. In our example, the sentence will be transformed to the expression shown in Example 2.6.4, where 'WHOM' is treated as a variable.

Example 2.6.4

```
((John present WHOM) with gift)
```

The resulting T-expression can then be subject to transformations by application of the S-rules, in an attempt to find a match with a T-expression in the knowledge base. If such a matching is found, the matching variables can be returned as answer to the question, and the sentences from which the matching T-expression were derived can be delivered as justification to the user.

2.7 AnswerBus

AnswerBus [Zheng, 2003] is an open-domain QA system which accepts questions in multiple languages (English, German, French, Spanish, Italian and Portuguese), translates them into English, and delivers answers in English only, extracted from web pages. An online demo is available at <http://answerbus.com/systems/index.shtml>.

After identifying the language of the question, the system translates it into English using BabelFish (<http://babelfish.altavista.com/>). The question is converted into an IR query, which is submitted to 5 different search engines.

A number of predefined question types are used to classify the questions. As an example, the category called 'DISTANCE' will be used for questions like *"How far"* and *"How close"*. When a numeric answer is expected, as in the examples mentioned, the unit of measure for the answer is decided on the basis of a predefined mapping of question words into units of measure.

The documents returned by the search engines are segmented into units, using in part the HTML formatting to identify paragraph and sentence boundaries. Among the resulting units, candidate passages are selected based on the degree of matching between the words they contain and the words in the query.

The degree of matching is also influenced by more specific factors such as the named entities contained in the candidate passage and the type of entity expected as answer by the question. Candidate answers with entities that match the question type will receive higher score. Coreference resolution also plays a role. A sentence contained a pronoun might be ranked higher if the resolution of that pronoun leads to a relevant passage. The score of a candidate answer will also be affected by the position of its source docu-

ment in the search engine ranking. A candidate answer from a document returned by a search engine with a better confidence score will get a higher score.

On the basis of the assumption that higher redundancy can be taken as positive support for a candidate answer, the answers which are found frequently will be scored higher. Finally, the system will present to the user a list of answers, ranked according to their relevance to the original question.

2.8 The ExtrAns system

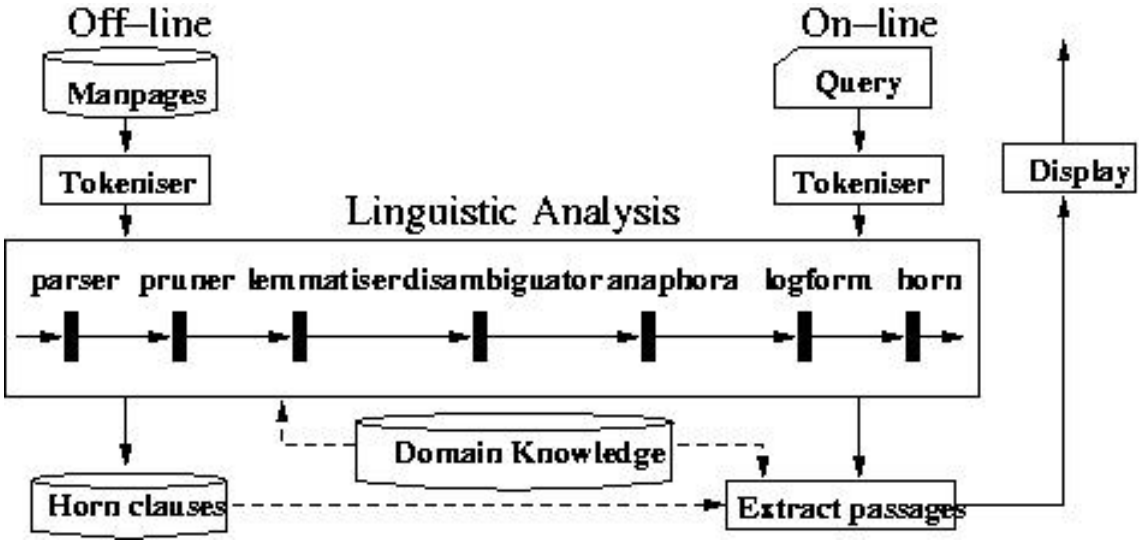
In this section we present a QA system (ExtrAns) developed at the Institute of Computational Linguistics of the University of Zurich in the years 1996-2000 [Molla, 2000], and provide examples from the initial application. We describe the ExtrAns system in detail, focusing in particular on the state of the system at the beginning of the work covered by this dissertation. Extensions which are part the author's work are described in Chapter 6, "Domain descriptions in the QA process".

2.8.1 Brief description

ExtrAns works by transforming documents and queries into a semantic representation called Minimal Logical Form (MLF) [Mollá et al., 2000a] and derives the answers by logical proof from the documents. A full linguistic (syntactic and semantic) analysis, complete with lexical alternations (synonyms and hyponyms) is performed. While documents are processed in an off-line stage, the query is processed on-line (see Figure 2.2).

Two real world applications have been implemented with the same underlying technology. The original ExtrAns system is used to extract answers to

Figure 2.2 . Architecture of the ExtrAns system (reproduced from [Molla, 2000])



arbitrary user queries over the Unix documentation files ('man pages'). A set of 500+ unedited man pages has been used for this application. An on-line demo of this early version of ExtrAns can be found at the project web page. [13]

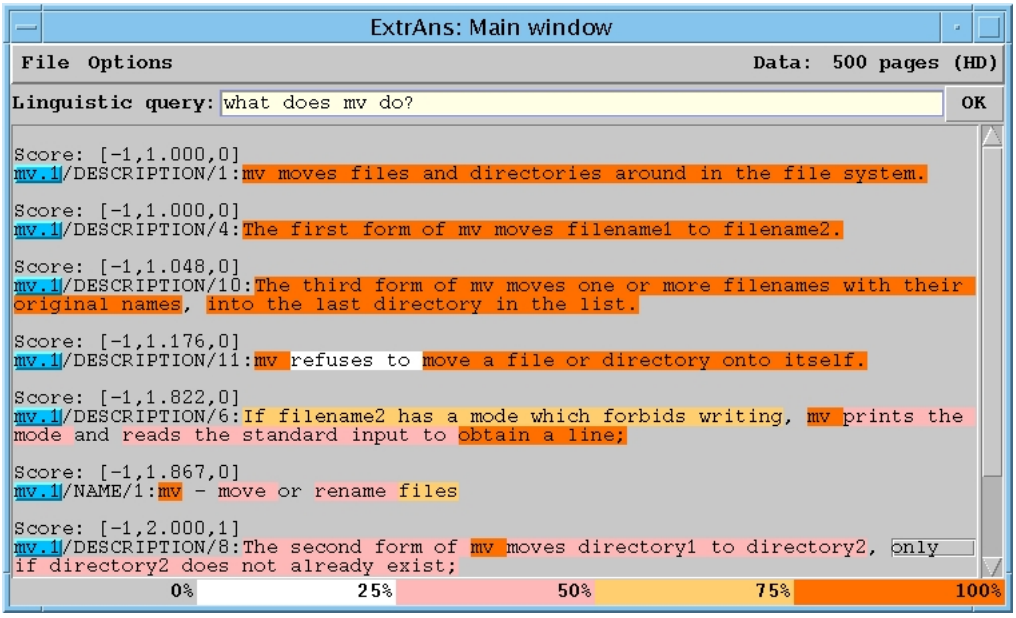
As ExtrAns works on relatively small volumes of data[14] it can afford to process (in an off-line stage) all the documents in the underlying collection rather than just a few selected paragraphs. Clearly in some situations (e.g. processing incoming news) such an approach might not be feasible and paragraph indexing techniques would need to be used. A preselection mechanism, based on a loose matching of question concepts against the stored semantic representations of the documents, is used to reduce the

[13] <http://www.ifi.unizh.ch/cl/extrans/>

[14] The size of the largest collection processed so far with ExtrAns is about 120 MB. This is about one order of magnitude inferior to the TREC collection (5 GB), and several orders of magnitude inferior to the estimated size of the text available on web (Terabytes).

complexity of the task. This approach is particularly targeted at small to medium sized collections. For larger collections an initial preselection module would be unavoidable.

Figure 2.3 . An example of the output of ExtrAns



In the following sections we describe in detail the various components of the system. Further information can be found in [Mollá et al., 2000a, Mollá et al., 2000b, Mollá et al., 2003a].

2.8.2 Lexical and Syntactic Analysis

The architecture of the ExtrAns system consists of several modules some of which are adaptations of third-party systems. As Figure 2.2 shows, the entire document collection is processed in an off-line stage but user queries are processed on-line. The same linguistic analysis is applied in both stages, transforming the input into a semantic representation called Minimal Logical Forms (MLFs).

Tokenisation is performed by a custom tokenizer, which has been specially

developed for each domain. In the case of manpages it needs to take into account the format peculiarities of unix troff as well as some conventions used in the manpages. In the case of Airbus it has been adapted to the XML markup used in the manual.

The syntactic analysis uses Link Grammar (LG) [Sleator and Temperley, 1993], which consists of a robust parser and a dependency-based grammar of English. The original output of LG has been extended to include the direction of the linkages as this information is important for anaphora resolution and MLF construction.

A corpus-based approach, inspired by [Brill and Resnik, 1994], is used to deal with ambiguities that cannot be solved with syntactic information only, in particular attachments of prepositional phrases, gerunds and infinitive constructions.

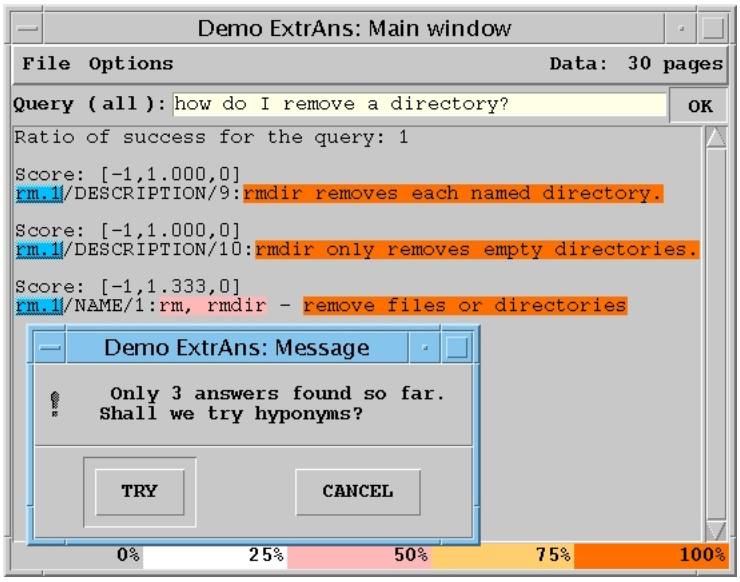
ExtrAns adopts an anaphora resolution algorithm [Lappin and Leass, 1994] that was originally applied to the syntactic structures generated by McCord's Slot Grammar [McCord et al., 1992]. So far the resolution is restricted to sentence-internal pronouns but the same algorithm can be applied to sentence-external pronouns too.

Nominalizations are dealt with using a small hand-built lexicon which takes care of the most relevant nominalized expressions. The main problem with derivational morphology is that the semantic relationship between the base words (mostly, but not exclusively, verbs) and the derived words (mostly, but not exclusively, nouns) is not sufficiently systematic to allow a derivation lexicon to be compiled automatically. Only in relatively rare cases is the relationship as simple as with 'to edit *a text*' $\longrightarrow$ 'editor of *a text*'/'*text* editor', as the effort that went into building resources such as NOMLEX [Meyers et al., 1998] also shows.

From the partially disambiguated dependency structures ExtrAns derives

one or more MLFs as semantic representation for the core meaning of each sentence.

Figure 2.4 . An example of the output of ExtrAns



Unlike sentences in documents, user queries are processed on-line and the resulting MLFs are proved by deduction over MLFs of document sentences. Pointers to the original text attached to the retrieved logical forms allow the system to identify and highlight those words in the retrieved sentence that contribute most to that particular answer [Mollá et al., 2000b, Mollá et al., 2000a]. An example of the output of ExtrAns can be seen in Figure 2.3. When the user clicks on one of the answers provided, the corresponding document will be displayed with the relevant passages highlighted. Another click displays the answer in the context of the document and allows the user to verify the justification of the answer. This is especially important in the case of procedural questions where an explicit solution to a problem is required.

2.8.3 Minimal Logical Forms

The success of ExtrAns depends heavily on its use of logical forms, which express the meaning of the documents and of the queries. ExtrAns' logical forms are designed so that they are easy to build and to use, yet expressive enough for the task at hand. Not least importantly, the logical forms and associated semantic interpretation method are designed to cope with problematic sentences. This includes very long sentences, even sentences with spelling mistakes, and structures that are not recognised by the syntactic analyser.

The ability of these minimal logical forms to underspecify makes them good candidates for NLP applications, specially when the applications benefit from the semantic comparison of sentences [Copestake et al., 1997, Mollá, 2001]. In the case of ExtrAns, the logical forms only encode the dependencies between verbs and their arguments, plus modifier and adjunct relations. Ignored information includes complex quantification, tense and aspect, temporal relations, plurality, and modality. ExtrAns' logical forms are called Minimal Logical Forms (MLFs), precisely because they ignore such more complex facets of the meaning of the sentences. The rationale for such simplification is that too detailed logical forms may interfere with the answer extraction mechanism. Additional information can be added incrementally to the minimal logical forms [Mollá et al., 2000a]. One of the effects of this kind of underspecification is that several natural language queries, although slightly different in meaning, produce the same logical form.

An important feature of the MLFs is the use of reification to achieve flat expressions (see Example 2.8.1). As opposed to Hobb's ontologically promiscuous semantics [Hobbs, 1985], where every predicate is reified, ExtrAns applies reification to a limited number of types of predicates, in particular

to objects, eventualities (events or states), and properties. That way we represent event modifiers, negations, higher order verbs, conditionals, and a finite number of higher order predicates.

The MLFs are expressed as conjunctions of predicates with all the variables existentially bound with wide scope. For example, the MLF of the sentence *"cp will quickly copy the files"* is shown in Example 2.8.1.

Example 2.8.1

```
holds(e4), object(cp,o1,x1), object(s_command,o2,x1),
evt(s_copy,e4,[x1,x6]), object(s_file,o3,x6),
prop(quickly,p3,e4).
```

In other words, there is an entity `x1` which represents an object of type `cp` and of type `command`, there is an entity `x6` (a file), there is an entity `e4`, which represents a copying event where the first argument is `x1` and the second argument is `x6`, there is an entity `p3` which states that `e4` is done quickly, and the event `e4`, that is, the copying, holds. The entities `o1`, `o2`, `o3`, `e4`, and `p3` are the result of reification. The reification of the event, `e4`, has been used to express that the event is done quickly. The other entities are not used in this MLF, but other more complex sentences may need to refer to the reification of objects (non-intersective adjectives) or properties (adjective-modifying adverbs).

ExtrAns' domain knowledge determines that `cp` is a command name, and the words defined in the thesaurus will be replaced with their synset code (here represented as `s_command`, `s_copy`, and `s_file`). For the Unix manpages domain a small domain-specific thesaurus based on the same format as WordNet [Fellbaum, 1998] was manually developed.

The MLFs are derived from the syntactic information produced by Link Grammar (LG) [Sleator and Temperley, 1993]. The methodology to produce the MLFs is relatively simple, one only needs to follow the main dependen-

cies produced by the LG. However, as it has been said elsewhere [Mollá et al., 2000a], the internal complexities of the dependency structures produced by the LG must be taken into account when producing the MLFs. The LG has a robust component that makes it possible to return structures even if the sentences are too complex or ungrammatical. The resulting structures can still be processed by ExtrAns and the corresponding MLFs are produced, possibly extended with special predicates that mark the unprocessed words as 'keywords'.

ExtrAns finds the answers to the questions by forming the MLFs of the questions and then running Prolog's default resolution mechanism to find those MLFs that can prove the question. Thus, the logical form of the question *"which command can duplicate files?"* is shown in Example 2.8.2 below.

Example 2.8.2

```
object(s_command,O1,X1), evt(s_copy,E1,[X1,X2]),
object(s_file,O2,X2)
```

The variables introduced in a question MLF are converted into Prolog variables. The resulting MLF can be run as a Prolog query that will succeed provided that the MLF of the sentence *"cp will quickly copy the files"* has been asserted. A sentence identifier and a pointer (indicating the tokens from which the predicate has been derived) are attached to each predicate of a MLF in the knowledge base. This information matches against additional variables attached to the predicates in the question (not shown in the example above) and is eventually used to highlight the answer in the context of the document (see Figure 2.3). The use of Prolog resolution will find the answers that can logically prove the question, but given that the MLFs are simplified logical forms converted into flat structures, ExtrAns will find sentences that, logically speaking, are not exact answers but are still relevant to the user's question, such as: *"cp copies files"*, *"cp does not copy a file onto itself"*,

"if the user types y, then cp copies files".

MLFs open up a potential path to a stepwise development of a question answering system by allowing monotonically incremental refinements of the representation without the need to destruct previous partial information [Schneider et al., 1999]. While MLFs specify the core meaning of sentences they leave underspecified those aspects of semantics that are less relevant or too hard to analyse for the time being.

2.8.4 Evaluation

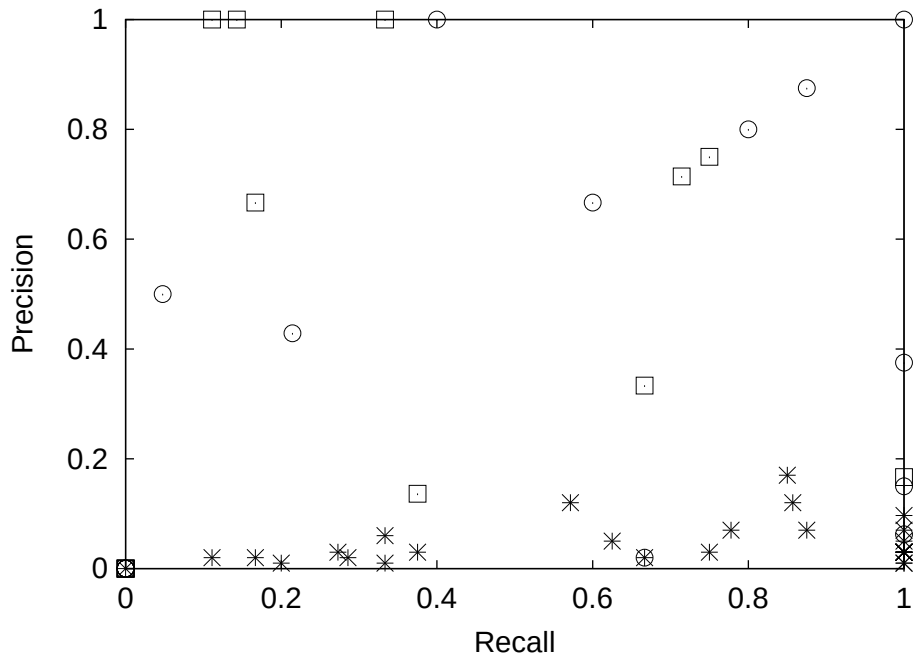
The evaluation described in this section was designed to compare the original ExtrAns system against a standard IR system. A set of 30 queries over 500 manual pages was used. The system chosen for the comparison was Prise, a system developed by NIST [Harman and Candela, 1989].

Since Prise returns full documents, we used ExtrAns' tokenizer to find the sentence boundaries and to create independent documents, one per sentence in the manual pages. Then Prise was run with our set of queries, which lead to an average of 908 hits per query.

The set of all correct answers was compiled mainly by hand. As Prise provides a ranked output, in order to compute precision and recall one has to select a cut-off value (`n`). The combined plot of pairs computed for each `n` did not show significant differences with the plot for `n=100`: the values for ExtrAns were nearly the same, and for Prise, the number of recall and precision pairs increased but the area with the highest density of points remains the same. We will therefore concentrate on the plot for `n=100`.

Figure 2.5 shows that precision is in general higher for ExtrAns than for Prise, and that Prise has better recall values. In the upper right corner, we can see a higher density of ExtrAns' values which is likely to shift to the left

Figure 2.5 . Recall against precision for 30 queries and the top 100 hits per query. Prise's results are displayed with a star (∗), and ExtrAns' results with circles (⊙) for the default search and with squares (□) for the approximate matching.} (reproduced from [Rinaldi et al., 2002c])



if we use a less restricted set of queries.

2.9 Conclusion

In this survey, after mentioning some historical precursors of today's QA systems, we described the current research efforts spearheaded by the TREC QA track, and some representative QA systems. Some of the problems that are common to all these systems are the lack of robustness of NLP parsing

and the difficulty of exploiting domain-specific knowledge.

While open-domain question answering systems (by definition) cannot make use of domain specific resources, restricted-domain systems can either attempt to leverage on existing domain resources, or can be enhanced with a domain model derived from the documents.

One possible strategy, which we will consider at length in the rest of this dissertation, is to make use of domain descriptions to extract the knowledge inherent in the documents (especially scientific and technical texts), and later make use of it in the question answering process.

Chapter 3

Domain Specific Collocations and Technical Terminology

One of the greatest hurdles in automatically processing scientific and technical documentation is the large number of domain specific multiword expressions (MWEs) that they contain. Such units typically describe domain specific entities (objects or procedures) that characterize the domain, and can be partly identified on collocational criteria.[1] In many technical texts, for example the Airbus Maintenance Manual (AMM) used as the document collection for our experimentation with the ExtrAns system, the names of different materials, parts of the aircraft, technician's tools and units of measure are so abundant within the documents, that without proper identification any NLP system would perform very poorly. In this work we have baptized them "**domain descriptions**".

Domain descriptions cover what might be called technical terminology, as well as linguistic items which are close to collocations proper to the scientific or technical domain to which the document belongs. Hence an item like *"tandem AP-1 consensus sequences"* is completely opaque to the layman, and

[1] I.e. as domain specific collocations.

falls within what we are calling here technical terminology, whilst a layman could, with some degree of confidence, guess at what was meant by *"hydraulically operated cargo compartment door"*. This latter is then an example of a collocation proper to the domain. However, both are domain descriptions, and both must be appropriately treated if we wish to improve the efficiency of any natural language processing components and to increase the chances of an appropriate response to a query being found.
The identification of domain descriptions in a pre-processing phase is beneficial for the following reasons:

- They can be used to increase the efficiency of parsing the documents and the queries.
- They can independently and additionally be used to mutually disambiguate other domain descriptions.
- The process of disambiguation leads to the discovery of domain relationships, which are helpful in the construction of a thesaurus.
- The thesaurus can be used to support inferences which are crucial for good question answering performance.

It is possible to draw a parallel between the pre-processing of domain descriptions that we propose in this work and the treatment of Named Entities (NEs) in classical open-domain question answering systems. Named entities (in the classical sense), include names of persons, organizations and locations. Typically NEs are recognized by a specialized component (NE extraction) and are reduced to a unique identifier, which allows the system to consider their possible variants as equivalent. For example, mentions of "President Bill Clinton", "Bill Clinton", and "Clinton" in the same document will be treated by the system as equivalent. Despite the parallel,

named entity extraction itself is not needed for technical documents, as they are unlikely to contain references to such entities. Conversely, a treatment of domain descriptions is not generally needed for processing news (the typical background documents in open-domain question answering), as they are not likely to appear frequently there.

3.1 Terms, collocations and other MWEs

Multiword expressions (MWEs) are a broad category which encompasses all multiword lexical units that somehow 'stick together'. It ranges from co-occurrences, to collocations, up to idioms. Co-occurrences are defined purely on the basis of statistical criteria, collocations on the other hand are defined on the basis of syntactic and semantic criteria. Idioms, at the extreme end of the scale, have a conventional meaning which is not necessarily related to their constituents. Among the MWEs frequently occurring in technical documentation we find technical terminology (the set of terms of the domain) as well as other domain specific collocations which describe objects of the domain.

Under a commonly used definition, terms are seen as unique descriptions of concepts. According to this point of view, they are unambiguous (one concept per linguistic expression) and canonical (one linguistic expression per concept). On top of that, they are generally opaque to the lay person.

On the other hand, expressions which denote domain objects without using domain-specific lexical items (and for this reason are transparent to a lay person) are also frequent. They are still (ideally) unambiguous, but they are not canonical, so that they are subject to a degree of variation. We will call these expressions "domain specific collocations". The union of technical terminology and domain specific collocations are what we call "domain

descriptions".

There might be of course linguistic items where it is unclear whether the item should be classified as a term or as a domain specific collocation: we have said elsewhere that in the last analysis only the judgment of a domain expert can decide these cases. Fortunately for us, defining a class of domain descriptions allows us to avoid this definitional issue, and it is further motivated by the fact that all elements of the class need to be treated in the same way for processing purposes. Thus, while a conventional treatment of terminology (in the strict sense) during parsing requires the use of domain specific resources such as terminology banks or glossaries, identifying domain descriptions can be done at least partially without prior knowledge of the domain - which is not to deny the potential utility of any domain specific resources as might exist.

One might say that domain descriptions and the relations between them encapsulate a certain amount of domain specific knowledge. In this sense, we are exploiting knowledge of the domain to improve the performance of the system, although we are not making use of any independently coded knowledge of facts about the domain. The technique we propose exploits linguistic properties of the text and of the items extracted from the text rather than relying on an external source of domain knowledge.

Domain descriptions may be single lexical units or multi word expressions. The latter, although apparently compositional, can be treated as single units on both a syntactic and a semantic level. On the syntactic level, treating them as units provides a considerable simplification of the parsing process, as we will show. On the semantic level, they can sometimes be replaced by a unique identifier (for example, a part number) which can be unambiguously interpreted by humans and machine.

3.2 The ISO definition

According to ISO 1087 [iso1087, 2000], a **term** is the *"Designation of a defined concept in a special language by a linguistic expression"*, whereas a **concept** is *"A unit of thought constituted through abstraction on the basis of properties common to a set of objects"*.[2] In other words, concepts are mental representations of the units of the domain (i.e. semantic objects), while terms are their possible names (i.e. linguistic objects).

If seen under this definition, domain descriptions are equivalent to the set of terms used to refer to objects of the particular domain, such as parts and tools in the case of a technical manual, or protein, genes, diseases, in the case of biomedical literature. However, this definition of a term is very strict, and whilst it fits well with the definitions used by professional terminologists, there are linguistic items in technical text which it would exclude, although they nonetheless feel closely akin to technical terminology, and although they benefit from the same sort of treatment as technical terms. It is not surprising, therefore, that there is a lot of controversy as to what is a proper definition of a term, and that the strict equivalence between term and domain description implied by the ISO definition cannot be guaranteed when a different definition has come into play.

The ISO definition of terminology quoted above is intensional in its nature and cannot be applied mechanically in order to determine the termhood of a linguistic item: it cannot be used to distinguish automatically terms from non-terms. Therefore some practical rules of thumb are often considered in trying to determine whether an item is or is not a term, in particular in the case of multi-word candidate terms, which are a subset of multi-word

[2] A recent survey of the epistemological status of the meta-terms 'term' and 'concept' in Terminology Theory can be found in [Kageura, 2002].

expressions (MWEs).

The rules traditionally used to establish 'termhood' include translational criteria and compositional criteria. The fact that a one-to-one translation of the constituents of a candidate term would not be an appropriate translation, is a good indicator of its termhood (translational criteria). Alternatively, if the meaning of the term can be identified by a lay person using a compositional interpretation of its constituents, this is a good indicator against its termhood (compositional criteria). Another way of stating this second criterion is to say that the sense of technical terms is opaque to the lay reader.

However these rules cannot be the sole criteria used to judge termhood. There are, for example, multi word expressions whose sense is not opaque to the lay reader, but which take on a flavour of termhood inside a particular domain. Arguably, there is no way to judge termhood on the basis of lexical or syntactical criteria alone. This is one reason why there are no reliable tools for terminology extraction. The final judgment on termhood always rests with a domain expert, who can distinguish lexical units which denote familiar objects of the domain from spurious candidates, which have no referent in the domain.

In our usage, domain descriptions cover both those linguistic items which would satisfy the traditional criteria for termhood and what we have called domain specific collocations, those multi word expressions which have a specific meaning in the domain being considered even though they are not in the conventional sense terms.

3.3 Domain descriptions and their relationships: previous research

There has recently been a surge of interest in the problem of structuring a set of domain descriptions. Given a set of domain descriptions, obtained from an existing resource or extracted from a corpus, the task consists in identifying various types of relations (e.g. synonymy, hyponymy) between them. Approaches that deal with the (semi-) automatic recognition of (near-) synonymy and hyponymy of domain descriptions can be broadly classified in three categories:

- exploiting information contained in the descriptions and their variations (*endogenous informations*)
- exploiting information provided explicitly by the authors in the text (*syntactically explicit information*)
- exploiting information inherent in the similar contexts of different, but closely related descriptions (*distributional similarity*)

3.3.1 Endogenous information

Various studies focus on variation in domain descriptions [Daille et al., 1996, Ibekwe-Sanjuan, 1998, Jacquemin, 2001] in order to expand an existing collection of descriptions over a given corpus. They typically identify different types of relations, with a clear focus on synonymy and hyponymy. Such approaches can be extended by distinguishing different "degrees" for each relation, corresponding to the different surface realization. For instance in the case of synonymy the idea is to distinguish between relations that induce "strict synonymy" (e.g. *"HIV gene expression* $\longrightarrow$ *HIV 1 expression"*)

from relations that induce "weak synonymy" (e.g. *"cytokine gene expression* $\longrightarrow$ *cytokine gene transcription"*). However, it is questionable whether such a clear-cut distinction can always be delineated.

[Nenadic et al., 2004] presents an approach to the automatic discovery of description similarities, which might serve as the basis for a number of knowledge mining tasks. They combine internal and external criteria. Lexical similarity is based on sharing lexical constituents (term heads and modifiers). Syntactic similarity relies on a set of specific lexico-syntactic co-occurrence patterns indicating the parallel usage of descriptions. Contextual similarity is based on the usage of descriptions in similar contexts. Other results in the Bio domain are presented in [Grabar and Zweigenbaum, 2004], where the authors apply their methods on the French version of the US National Library of Medicine MeSH thesaurus.

[Navigli et al., 2005] describes a qualitative evaluation of the OntoLearn system, which uses Terminology Extraction and Semantic Interpretation of domain descriptions in order to extend an existing seed Ontology. [Pum-Ryu and Key-Choi, 2005] presents ways to measure the specificity of a description, and describes experiments based on a selected sub-tree of the MeSH thesaurus. [Marie-Reinberger and Spyns, 2005] makes use of dependency relations to cluster domain related descriptions, which are then evaluated against the UMLS. [Cimiano et al., 2005] presents an approach which combines heterogeneous sources of evidence and exploits different learning paradigms. [Nédellec and Nazarenko, 2005] discusses the complementary role of Information Extraction and Ontology, where IE can be used to populate an Ontology, or the Ontology can be used as a support for the IE task. Their approach is illustrated with examples from the biology domain.

3.3.2 Syntactically explicit information

Approaches that rely on explicit syntactic information provided by the author have been proposed [Hearst, 1992, Morin and Jacquemin, 2004]. Patterns that can be used to identify NP_1 (and possibly further conjoined NPs) as an hyponym of NP_0 include:

- such NP_0 as NP_1; NP_0 such as NP_1; NP_0 including NP_1;
- NP_0 and other NP_1; NP_0, especially NP_1; NP_0, for example NP_1;
- identify NP_0 as a NP_1; some NP_0 (NP_1, NP_2, ...)

The class of patterns for nouns that are not necessarily hyponymous but loosely related is quite large, for example:

- NP_0 and NP_1; NP_0 or NP_1; NP_0 is NP_1 (and other copular verbs)

The advantage of such "cue-phrase" methods is that they are quite easy to implement and they have high precision. But their recall is relatively low because the hyponymy patterns they rely on can be quite infrequent. [Morin and Jacquemin, 2004] report 82% precision and 56% recall for the hypomymy recognition task. If the amount of text available is restricted – which is usually the case in a specific domain – low recall becomes a problem. If the broader class of loosely related nouns is included, recall increases, but often at an unacceptable decrease in precision.

Linguistically it can be said that the *syntagmatic* relations between NP_0 and NP_1 contained in the hyponymy patterns explicitly express hyponymy: they directly link a cue-phrase to a semantic relation. The vast majority of lexical relations, however, do not express hyponymy or leave it unclear. How are we to know from "*NP_0 loves NP_1*" if there could possibly be a synynomy, hyponymy or semantic closeness relation between NP_0 and NP_1?

3.3.3 Distributional similarity

What we do know, however, is that all NP_0 in *NP_0 loves NP_1* need to be able to experience feelings, hence are animate in non-metaphoric uses. The *love* relation places a semantic restriction on its subjects. Linguistically, there is a *paradigmatic* relation between all the potential NP_0. Entities that appear in the same relations and with comparable frequency in texts are semantically related in a loose way. This was formulated by [Harris, 1968] as the *distributional hypothesis*: "The meaning of entities, and the meaning of grammatical relations among them, is related to the restriction of combinations of these entities relative to other entities."

Distributional similarity has been used to discover noun classes [Hindle, 1990], cluster similar words [Lin, 1998], do word-sense disambiguation [Schütze, 1998] and discover synonyms [Linden and Piitulainen, 2004].

The underlying intuition in distributional similarity is that two words are distributionally similar if they appear in similar contexts. The notion of context can be interpreted in many different ways. It can be neighbouring words, a window of some words to the left and the right, an entire document or a semantic or syntactic relation. In the example of a syntactic subject relation as context, the NP_0' that tends to occur most often with the same verbs as NP_0 is its best synonym.

All nouns and all the verbs with which they have subject relations are cross-tabulated, say nouns as rows and verbs as columns. This is a so-called vector-space model. A cell in this table is the count of a given noun in the subject relation to a given verb. Instead of the count, a value weighted by the relative frequency of the verb or a saliency-based metric such as TFIDF is typically used. For a given noun row, a so-called vector, the noun row that is least different is the closest synonym.

There are a number of measures for the distance between two vectors. The most simple ones are edit distance (how many cells are different?) or block metric (what is the total sum of the differences between cells?). A variant of the latter, weighted by relative verb frequency, is the Minkovsky or `L_1` measure [Dagan et al., 1999].

$$L_1(N_0, N_1) = \sum_{verb} \mid P(verb \mid N_0) - P(verb \mid N_1) \mid \tag{3.1}$$

Like the equally popular information theory measures (e.g. MI, [Lin, 1998]) `L_1` is symmetric, i.e. `L_1(N_0,N_1) = L_1(N_1,N_0)`.

[Weeds, 2003] suggests a non-symmetric distance measure, *co-occurrence retrieval*. It is based on the question of what happens if one replaces `N_1` for `N_0` in a text, of how often such a replacement is "wrong". Suppose that `N_0` is the gold standard vector to be compared to a retrieved `N_1` vector, whose correctness we need to evaluate. `N_1` can be "incorrect" in two ways: either a verb predicted by `N_0` is actually not present for `N_1` (a precision error), or a verb expected by `N_1` is not found in `N_0` (a recall error). [Weeds et al., 2005] shows that this recall measure is especially efficient at detecting synonymy relations: the higher the recall between two nouns, the more closely related they are. An evaluation on the GENIA ontology shows that two neighbour nouns according to the recall measure are in the same lowest (the most specific) level in the ontology in 93.4% of the cases [Weeds et al., 2005].

In addition to detecting synonymy, this method can be used to predict classes, by grouping sets of closest neighbours. The reported accuracy for assigning semantic classes is, over the GENIA ontology, 63% for all classes, and 76% for classes containing at least 5 members.

3.4 Conclusion

In this chapter we have introduced the notion of domain description and we have discussed its relation with more standard terminology and with domain specific collocations. We have then surveyed previous work by other authors aimed at extracting relationships among domain descriptions from document collections. Unfortunately, it is common to confuse the notions of domain specific collocations and terminology proper, so that many authors do not make this distinction.

The research activities that we have surveyed in the previous section typically aim at the construction of a domain model for direct inspection. We are not aware of other works which have the aim of making use of the derived structures within a complex NLP system, such as a question answering system.

In order to make effective use of domain descriptions in a QA system, it is necessary to be able to solve a number of related problems:

- identify the descriptions in the source documents and in the user queries
- recognize their interrelationships (which reflect the structure of the domain)
- apply them appropriately at the syntactic and semantic levels of processing

In the following three chapters we show how domain descriptions were detected in the technical maintenance manual of a well-known commercial aircraft, and how they were used within an extension of the question answering system previously described in Section 2.8.

In Chapter 4, "Extraction of Domain Descriptions" we present and evaluate the techniques that we have used for the extraction of domain descrip-

tions from the technical manual. The domain descriptions are also subject to variability in their instantiation. Morphological variability is an obvious example, but synonymy, use of acronyms or changes in word order may also be sources of variation. All of the different ways a single concept may be referred to in the domain are gathered into synonymy sets (Chapter 5, "Structuring of domain descriptions"). Creating the taxonomy and grouping together descriptions into synonymy sets amounts to building a (partial) domain ontology, which can later be exploited in an NLP system. The usage of the structured set of domain descriptions within our question answering system is discussed in Chapter 6, "Domain descriptions in the QA process".

Chapter 4

Extraction of Domain Descriptions

Because of their relatedness with terminology proper, one might hope that the same procedures that are proposed for the process of terminology extraction can be applied in a similar fashion to the extraction of domain descriptions. Terminology extraction tools typically are very limited. What they do, in general, is to rely on statistical information (collocations) or minimal lexical information (PoS tagging), in order to extract a ranked list of noun phrases. They differ mainly in the ranking criteria used to sort the list of candidates that they generate, which is then offered to the user for manual selection.

Nonetheless, because of the relationship between terminology and domain descriptions, it might be thought useful to adopt the same approach for the extraction of domain descriptions. The only difference is in the instructions given to the domain expert in charge of making the final selection among the candidate terms (or descriptions). In this section we describe some techniques for terminology extraction, which we used to extract domain descriptions from the Maintenance Manual of the Airbus A320 (AMM).

4.1 Strategy

Existing terminology extraction tools typically use statistical collocational measures in order to extract domain specific collocates as term candidates. As they have no domain-specific knowledge, they typically target recall at the expense of precision, therefore requiring a phase of manual validation of the candidate terms [Castellví et al., 2001]. This might due to the inherent difficulty of capturing in a domain-independent fashion the very notion of termhood. In the case of unstructured document, they can be helpful anyway in order to support the process of extraction of domain descriptions. However, where partially structured documents are available (for example XML annotated documents), it might be possible to leverage on the document structure to support the process of identification of domain descriptions.

We constructed a validated list of domain-specific descriptions from different external and internal sources. The Aircraft Maintenance Manual made available to us is divided into 46 chapters. We concentrated our attention on the three chapters most often queried by technicians, which cover approx. 1 million words, making up 10% of the full manual. Two separate automatic extraction methods were used, as we describe in the next section. The results were combined and manually validated. We ended up with a list containing approx. 13000 descriptions and variants (including spelling and morphology).

In section Section 4.2 we describe how the three chapters of the AMM have been fully analyzed and domain descriptions have been semi-automatically extracted and manually verified (using the visualization tools described in Section 4.3). Considering the extracted list to be relatively complete in terms of the selected chapters enabled us to explore different features of both the

analyzed document and the descriptions themselves. This list served as a basis for an evaluation of automatic term extraction tools and methods (Section 4.4), in particular regarding their potential usage as extractors of domain descriptions. Additionally, we discuss some properties of the extracted descriptions (Section 4.5).

4.2 Methodology

In the framework of the research reported in this thesis, two separate approaches were considered and evaluated for extracting domain descriptions from the AMM manuals. Different sources of information, both internal and external, were invaluable in the extraction process. First, several kinds of external sources (glossaries of abbreviations used in aircraft industry and different technical specifications, e.g. [ATA, 1997]) were used. Secondly, different types of structures in AMM can indicate the presence of a description, and thus be exploited. Some of the descriptions are already explicitly denoted through the use of markup (e.g. element `CONNAME` for consumable material, element `TOOLNAME` for tools etc) (Section 4.5), and this annotation also proved to be a valuable resource.

The first approach to extraction is based on a stop-phrase method that splits up the content of certain SGML-zones (titles, paragraphs) using a list of units (nouns typically) that often hint at the presence of an adjacent description. For example, from a task title *Check of the Electrical Bonding of External Composite Panels with a CORAS Resistivity-Continuity Test Set* we cut out stop-phrases like *of the*, *of*, *with a* to obtain a list of candidate descriptions: Check, Electrical Bonding, External Composite Panels, CORAS Resistivity-Continuity Test Set. Given the high density of descriptions in the material we are dealing with, even such a crude method can provide interesting re-

sults.

A second approach that we considered is a fully automatic statistical method [Dias et al., 1999]. The method is general, using no linguistic analysis, allowing n-grams to be of any length and allowing them to be non-contiguous (i.e. they can contain "holes"). It uses *Mutual Expectation* as an association measure, which evaluates the cohesiveness of a multi-word unit and a criterion called *LocalMax* to select the candidate terms from the evaluated list.

This method was implemented as a Perl program and applied to the three chapters of the manual, covering approx. 1 million words (10% of the complete text). The SGML markup was removed from the text beforehand.

Due to the low precision of the result, two stop-word filters were used to prune the output, which either removed the suspect extracted units from the list or modified them by deleting words from the beginning and end. This kind of stop-word filtering reduced the size of the list by about 30%.

4.3 Visualization Tools

In order to simplify the manual verification and correction (either pruning or supplementing) of the extracted descriptions, specific visualization tools were developed. The need for good visualization tools is motivated by the typical behavior of terminology extraction tools, which often produce high recall at the expense of precision.

As a preliminary step, the original SGML format of the manual was converted into XML[1]. Using standard off-the-shelf tools we developed a simple XML-to-HTML converter that allows us to inspect the manual using a conventional browser. It is extremely helpful to be able to visualize the ex-

[1]With some loss of information, though not relevant for our application (mainly related to versioning of the original document).

tracted descriptions in the context where they appear. In order to achieve this, additional XML markup denoting the extracted units is inserted into the manual.

The new markup tags can be tied to presentational information (defined by CSS stylesheets, for example), so that when the manual is browsed the descriptions are highlighted and differentiated from the rest of the text. All modern web browsers are capable of handling such specifications. Figure 4.1 is an example of how the identified descriptions are presented in a different colour and with additional bracketing in the context of the manual.

Even a preliminary understanding of the internal structure of the descriptions (and therefore possible hyponym-hyperonym relations) is made possible using a visualization based on nested levels of markup. For example, the multi-word unit *"flight crew electrical foot warmers"* is marked in the following way (where the inner levels of nesting can be presented with darker color etc):

```
[[flight crew] [electrical [foot warmers]]]
```

We discuss in the next chapter (Chapter 5, "Structuring of domain descriptions") how the structural information can be gathered from the complete list of domain descriptions.

4.4 Evaluation

Manual validation, making heavy use of the visualization tools, was used to produce a complete list of the domain descriptions present in the three chapters of the AMM serving as our document collection. This list was then used as a gold standard against which automatic extraction tools could be assessed.

The list obtained by the statistical method of Mutual Expectation (ME) and LocalMax (combined with simple stop-word filtering) gave results with 44% recall and 15% precision. For the list obtained by the stop-phrase method the recall was 66% and precision 12% [Rinaldi et al., 2002a]. Better recall results can be explained by the fact that the stop-phrase method is aware of the structure of the manual and "knows" how important information and descriptions are presented there, while the statistical method is general and makes only use of the frequencies and cohesiveness of the multi-word units. When combining the methods, the recall grew to 78% and precision fell to 10% [Rinaldi et al., 2002a]. The two methods produced lists of descriptions with relatively small intersection (only approx. 2000 items). Although we do not have a final explanation for this surprising result, we suspect that it might be related to the completely different approach towards extraction that the two methods take. While one of them makes use of structural information provided by the authors of the documentation, the other makes use of purely statistical properties of the data.

We conclude that these methods are useful for obtaining a preliminary set of domain descriptions, which can then be visualized to help manual checking. Even the statistical method which showed a low result of recall is still valuable for backing up the stop-phrase method.

4.5 Properties

The main test-set for our experiments was a combined list of terms from the three chapters of the manual, which together contain approx. 1 million tokens. Besides evaluating the two automatic terminology extraction methods in the hope of finding the most suitable for the AMM, we also analyzed whether the results obtained are general enough to be considered represen-

tative of the whole manual.

The list of descriptions extracted from the selected chapters contained approx. 13000 descriptions of which approx. 1000 were single-word tokens.

As we tried to extract all the possible spelling and morphological variants of the descriptions, we also grouped them so that each group contained all the spelling variants of one description. This kind of grouping reduced the size of the list by 20%.

Unfortunately, the chapters share very few descriptions with each other; only about 250 descriptions are present in all three chapters, about 550 are present in two, the rest appear in only one chapter. This is not a surprising result as different chapters of the manual tend to be thematically different.

Looking in detail at one of the chapters, 58% of the descriptions were only present there, 24% of them were shared by 2 to 4 chapters, and only 18% were a bit more general. The domain descriptions tend to be chapter-specific — the statistics show that each chapter is likely to contain its own unique terminology, which means that no chapter can be ignored in the process of extracting the descriptions. This is not a surprising result, given the nature of the documents that we analyzed, where different chapters tend to focus on different topics.

A great number of domain descriptions appear in the manual only once (i.e. they are "hapax legomena"), which means that detecting them by frequency based methods is likely to fail.

According to our results, most of the descriptions are multi-word units, usually bigrams and trigrams, but in principle there is no limit to the number of tokens they can contain.[2] The result of the selection process does (optionally) also include placards and messages, which tend to be particularly long, as in Example 4.5.1. Examples of domain descriptions proper (domain spe-

[2]Here by 'token' we mean a string of characters bordered by either a space or hyphen.

cific collocations referring to domain objects) are shown in Example 4.5.2.

Example 4.5.1

- *USA MIL-S-81733 CLASS C CORROSION INHIBITIVE INTERFAY SEALANT*
- *system status message 'SLIDES PRESS LOW'*

Example 4.5.2

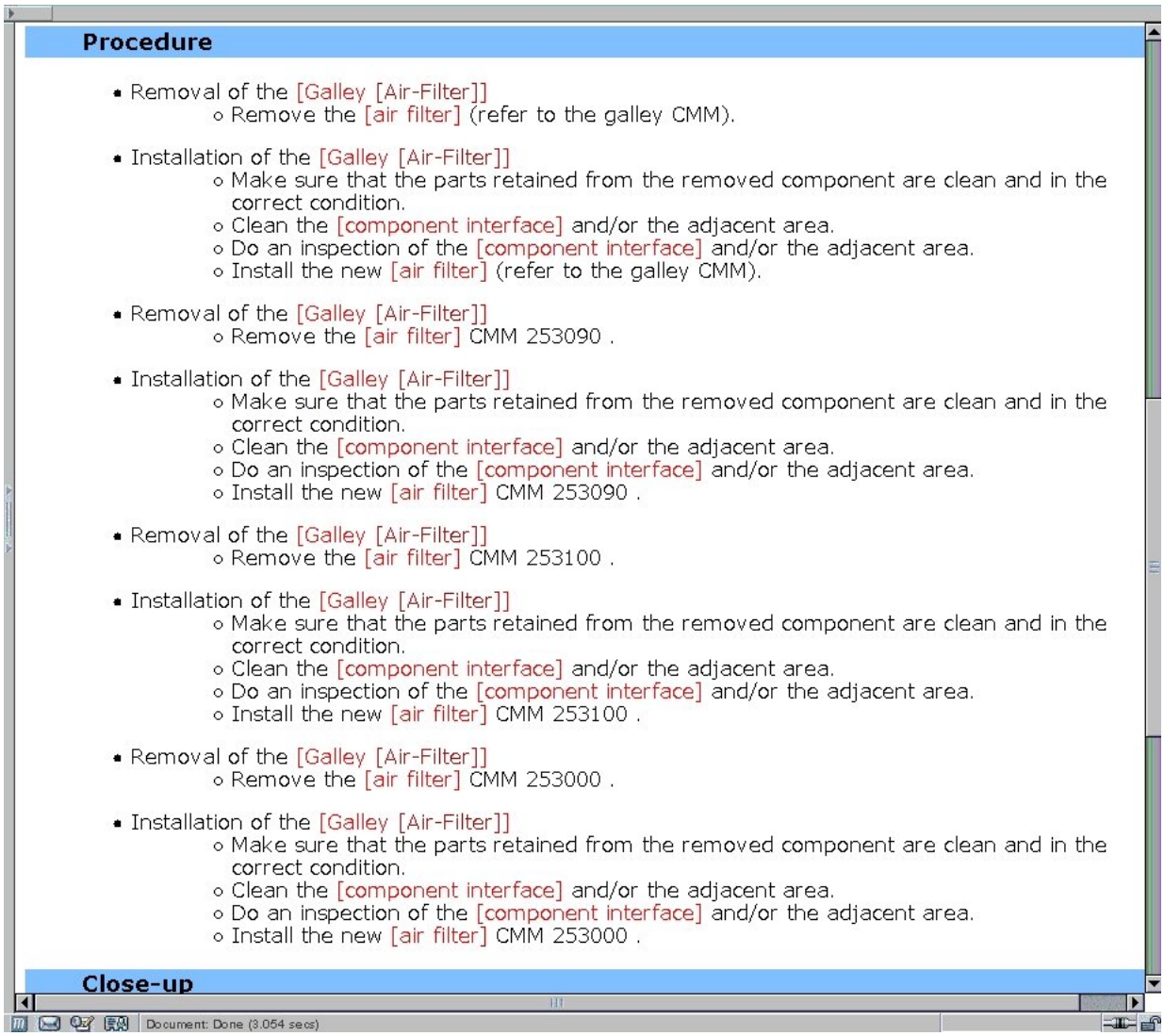
- *bleed pressure regulator valve control solenoid*
- *hydraulically operated cargo compartment door*
- *flight crew electrical foot warmers*

Bigrams and trigrams account for approx. 80% of the total number of multi-word terms. There are however a few cases of terms involving as many as seven or eight tokens.

4.6 Conclusion

In this chapter we have presented techniques that were adopted for the extraction of domain descriptions from the AMM manual. Two different extraction methods have been presented and evaluated. Although neither of them is sufficiently advanced to allow a fully automated extraction, they do provide much needed support in the process of identification of the domain descriptions. However, the descriptions alone are of little use without the knowledge of their interrelationships. The discovery of some of such interrelationship is the topic of the next chapter.

Figure 4.1 A conventional browser used to inspect terminology (reproduced from [Rinaldi et al., 2003b])



Chapter 5

Structuring of domain descriptions

Despite all efforts in standardization, it is often unavoidable that different writers and editors use different (but related) surface forms[1] to refer to the same domain concept. Even in consolidated sectors (like the aircraft industry) there are no absolutely reliable methods to enforce standardization across different editors. Besides, new technical developments lead to the continuous creation of new descriptions to refer to novel domain concepts. Consequently, when processing technical documents, it is vital to recognize not only standardized descriptions (as obtainable for example from a parts list) but also potential variations and possible new descriptions.

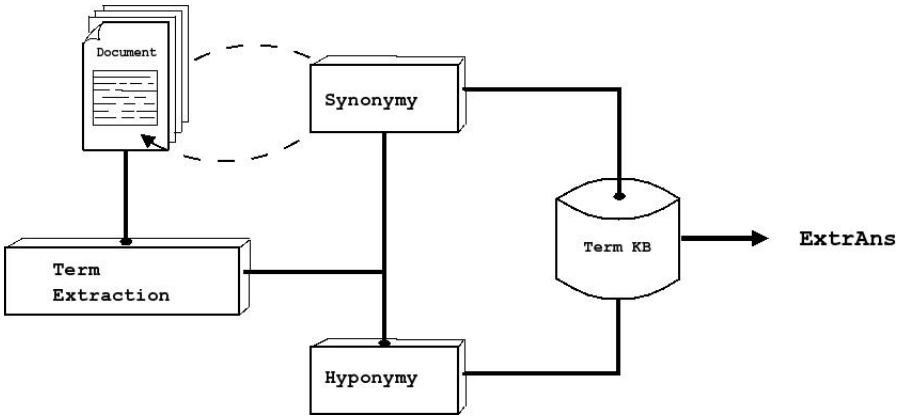
The process of terminological variation is well investigated [Ibekwe-SanJuan and Dubois, 2002, Daille et al., 1996, Ibekwe-Sanjuan, 1998, Jacquemin, 2001]. The primary focus of research has been to use linguistically based variation to expand existing term sets through corpus investigation in order to collect an exhaustive set of domain descriptions, or to produce domain representations for visual inspection.

We have applied similar techniques to the list of domain descriptions ob-

[1] Which we have called *descriptions*, see Chapter 3, "Domain Specific Collocations and Technical Terminology".

tained from the AMM. We describe below how such techniques can be used to detect synonymy and hyponymy.

Figure 5.1 Processing of Domain Descriptions.



5.1 Synonymy

In order to find synonymous descriptions we adapted the tool FASTR [Jacquemin, 2001] by adding metarules, as described below. However, a preliminary stage is necessary in order to detect simple orthographic variants and acronyms. We first normalize any description that contain punctuation by creating a punctuation free version and recording that the two are strictly synonymous. Further processing is involved in descriptions containing brackets to determine if the bracketed token is an acronym or simply optional. In the former case an acronym-free description is created and the acronym is stored as a synonym of the remaining tokens which contain it as a regular expression. So *evac* is synonymous with *evacuation* but *ohsc* is synonymous with *overhead stowage compartment*. In cases such as *emergency (hard landings)* the bracketed tokens cannot be interpreted as an acronym and so are not removed. [2]

[2]The problem of identifying acronyms in technical documents, and relating them to

FASTR, originally designed as a terminology extraction tool, is capable of identifying variants of known syntactic units in a given text. Although the author suggests that the main purpose of his work is terminology extraction, the tool itself makes uses of a general approach (described below), which is not necessarily restricted to terms. In fact, it could be claimed that the syntactic units that FASTR operates upon are more likely to be domain specific collocations than terms proper. However, in this chapter, to avoid confusion, we will assume that it can operate more generally on domain descriptions.

The individual words involved in a previously extracted base set are associated with their part-of-speech[3], their morphological root[4] and their semantic synset as defined by WordNet[5]. How tokens combine to form multi-token descriptions is represented as a phrasal rule, the token specific information carried in feature-value pairs. Metarules license the relation between two descriptions by constraining their phrase structures in conjunction with the morphological and semantic information on the individual tokens.

Therefore synonymous descriptions can be reliably identified on the basis of the concurrent internal and external evidence. The internal evidence is generated by the metarules, which show how one description can be generated from another. The external evidence is provided by their cooccurrence in a given text unit.

Using phrase structure rules in combination with a morphological database

their full forms, is a major technical issue. For example a very interesting approach to this problem in the domain of biomedical literature is presented in [Okazaki and Ananiadou, 2006]. We have limited ourselves to very simple techniques, based on the length and orthographic structure of the bracketed tokens.

[3]assigned by the IMS TreeTagger, see <http://www.ims.uni-stuttgart.de/projekte/corplex/TreeTagger/>

[4]obtained from CELEX, see http://www.kun.nl/celex

[5]http://www.cogsci.princeton.edu/wn

and WordNet, linguistic variations between two descriptions can be identified. This can detect simple head inversion (*water flow* $\longrightarrow$ *flow of water*), morphological variations (*electric connector* $\longrightarrow$ *electrical connector*) and complex morphosyntactic variations (*electrical generation equipment* $\longrightarrow$ *equipment for generating electricity*).

Exploiting the WordNet synsets allows weaker synonymy relations to be discovered. Descriptions with synonymous heads (*bulk cargo* $\longrightarrow$ *bulk load*), synonymous modifiers (*upright position* $\longrightarrow$ *vertical position*) or both (*functional test* $\longrightarrow$ *operational check*) are all detected.

We designed the metarules to identify strict synonymy among descriptions, which results from morpho-syntactic variation (*cargo compartment door* $\longrightarrow$ *door of the cargo compartment*), descriptions with synonymous heads (*electrical cable* $\longrightarrow$ *electrical line*), descriptions with synonymous modifiers (*fastener strip* $\longrightarrow$ *attachment strip*) and both (*functional test* $\longrightarrow$ *operational check*). For a description of the frequency and range of types of variation present in the AMM see [Rinaldi et al., 2002a].

Syntactic variations involve either inserting an argument (word or acronymn) into an existing description (Example 5.1.1) or permutating an existing syntactic structure (Example 5.1.2). Although some of the examples reported might not appear to be perfectly synonymous at first sight, we observe that, in the context of the technical manual, they are used as such. The technical editors have, in the process of writing the manual, spontaneously made use of a modified description which, in the correct context, can be interpreted unambiguously.

Example 5.1.1

galley electrical system $\longrightarrow$ *galley power electrical supply system*

cargo door $\longrightarrow$ *cargo compartment door*

polycarbonate sheet $\longrightarrow$ *polycarbonate (pc) sheet*

Example 5.1.2

seat assembly $\longrightarrow$ *assembly of each seat*

water flow $\longrightarrow$ *flow of water*

This sort of variation is relatively productive accounting for 33% of the description instances in the corpus. However, these simple syntactic variations are also involved in conjunction with types of morphological and semantic variation.

Purely morphological variations exchange morphologically related words. This type of substitution combined with a syntactic insertion (Example 5.1.3) is rare (only two occurrences), more common is a combination with a permutation on the original syntactic structure. Morphologically centered variations make up 25% of the indexed variants.

Example 5.1.3

electrical equipment $\longrightarrow$ *electrically operated equipment*

Example 5.1.4

bulk cargo $\longrightarrow$ bulk load

30% of the variants are semantic in nature. As with the morphology, a simple variation substitutes words. For a semantic variation the words must belong to the same synset, either heads or modifiers (Example 5.1.4). Semantic substitution can be combined with insertion to define the relation in Example 5.1.5. The variant in Example 5.1.6 is a permutation of *cargo door* and is related to the base description as *load* and *cargo* belong to the same synset.

The remaining variations (12%) were in punctuation and orthography. These arbitrary differences represent strictly synonymous descriptions whereas (Example 5.1.6) is the weakest useful synonymy relation, and there are as

Example 5.1.5
minimum distance ⟶ minimum handling space

Example 5.1.6
load door ⟶ door for the cargo

many differing degrees between these two extremes as one cares to discover. Where some studies [Hamon and Nazarenko, 2001] focus on these degrees, we take a conflation approach to these relations. As such, all variants of a single concept are grouped in WordNet type synsets.

5.2 Hyponymy

The availability of additional relationships among domain descriptions can support the process of question answering, by allowing the system to find answers that contain descriptions which are not necessarily strictly equivalent, but nevertheless related to the entities that the user mentions in his questions. In particular, the relation of hyponymy is useful for inferencing purposes, as it allows expanding a query containing a given domain description with all its hyponyms, thus providing a richer set of answers, still logically correct.

The relation of hyponymy among domain descriptions can be derived from knowledge of their internal structure. Unfortunately, these expressions are syntactically many-fold ambiguous, and this structural ambiguity is unresolvable from consideration of the description alone.

For example, an *adjustable access platform* is an *access platform* which is *adjustable*. However, a *crew member seat* is a *seat* for a *crew member* and an *underfuselage off-centered door* is a *door* that is both *underfuselage* and *off-centered*. Moreover, the compositional ambiguity that we are trying to resolve is only

one of the ambiguities that are inherent in a nominal compound. In fact, in any compound of the form (N N), the nature of the relationship between the two words cannot be easily detected. For example, *"escape slide"* is a slide for the escape, but a *"printer output"* is output made by a printer, and *"syntax analysis"* is analysis of the syntax, and so on.[6]

However, since there is a systematic connection between the syntactic structure of a description and the semantic relationships among the concepts denoted by its constituents, we do not need to consider either of these types of ambiguity. In particular, the relation between the head and the full compound corresponds to a semantic hyponymy relation among the corresponding concepts. It is, in other words, a general linguistic fact that the relationship between a nominal compound and its head remains constant across these possible variants. For example, a *"crew member seat"* is a kind of *"seat"* regardless of the internal bracketing of the compound.

While for the detection of hyponymy, in the sense described above, the internal structure of the compound would be irrelevant, knowing its internal structure is needed for detecting other classes of domain relations. For example, if we know from other sources that a *"pilot"* is a *"crew member"*, then the knowledge of the internal structure of *"crew member seat"* can lead us to the conclusion that a *"pilot seat"* is a *"crew member seat"*.

Syntactically, the possible structures allowed for *"crew member seat"* are the following:

- `[[crew member] seat]`
- `[crew [member seat]]`
- `[[crew] [member] seat]`

[6]This is a problem that we have chosen to ignore within the present dissertation.

However, the existence of the description *crew member*, together with the absence of **member seat*, suggests that only the first structure is correct.

Similarly, in the case of *adjustable access platform*, the existence of the hypernym parent *access platform* as a description, suggests that the more specific description is created by simply modifying the parent to be *adjustable*: `[adjustable [access platform]]`.

And finally, the lack of any more specific description (or the existence of both of them) results in the individual modification of the single head, as in: `[ [underfuselage] [off-centered] door]`.

In other words, it is theoretically possible to segment a multi-word description in a number of ways but only some segmentations result in segments which exist independently as domain descriptions in their own right. Finding a segment which is itself a domain description can validate a possible disambiguation of the larger multi-word domain description of which the segment is a part.

Being able to detect the internal structure of the domain descriptions provides therefore a considerable support in the process of construction of a domain representation, which can be useful for various applications, including question answering.

While each description in isolation cannot be disambiguated, the consideration of the entire set of domain descriptions can lead to a decision procedure which is surprisingly effective, as we show below. The core idea is based on the observation that, in case of incorrect bracketings, the constituents would denote entities which actually do not exist in the domain. Therefore such partial constituents cannot occur independently among the set of descriptions. So the existence (or non existence) of the constituents as separate descriptions is a strong indicator for a correct (or incorrect) bracketing.

More formally, in order to detect hyponymy relations among domain de-

scriptions, the following steps are taken. The domain descriptions are tokenized (removing some differences in punctuation, such as hyphens), lemmatized (removing some trivial distinctions such as singular vs plural), and tagged. After removing duplicates, the remaining descriptions are compared pairwise to check if they properly subsume each other. A description `A` is assumed to be an hyponym of description `B` if the three conditions below are satisfied:

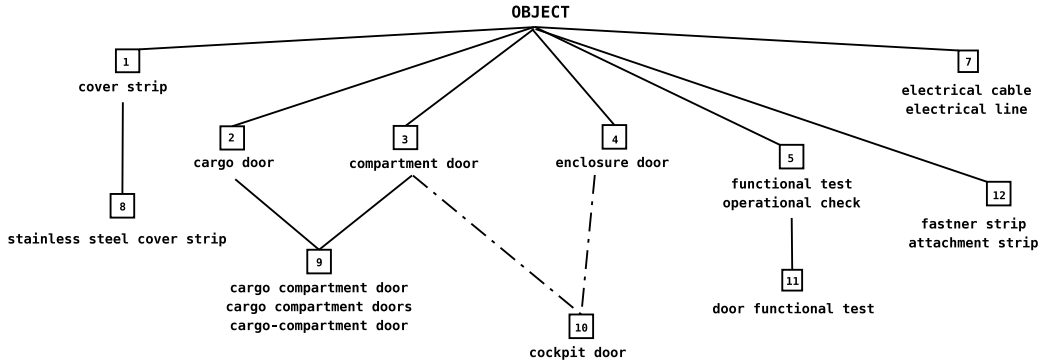
- `A` has more tokens than `B`
- all the tokens of `B` are present in `A`
- both descriptions have the same head

In the vast majority of cases the rightmost token can be assumed to be the head of the description, however there are a few cases where it is necessary to rely on the tagging information in order to identify the head. So, more generally, the rightmost noun is taken to be the head of the compound.
Automatically discovering these thesaurus relations across 6032 descriptions from the AMM produces 2770 synsets with 1176 hyponymy links. Through manual inspection of 500 synsets 1.2% were determined to contain an inappropriate description. A similar examination of 500 hyponymy links verified 96% of them as valid.

5.3 Conclusion

The information derived from the discovery of the synonymy and hyponymy relations was used to create a knowledge structure which we have elsewhere called a "computational thesaurus" [Rinaldi et al., 2002a,Rinaldi et al., 2003b] or a "terminological knowledge base" [Rinaldi et al., 2003c]. Figure 5.2 shows graphically a fragment of the resulting structure, which is

Figure 5.2 A sample of the AMM computational thesaurus.



structurally similar to WordNet. Synonyms are conflated within a single unit (called a 'synset', in analogy to WordNet), which is represented by a unique symbolic identifier (numbers within a square). The relation of hyponymy is represented by the lines connecting the synsets. Given that the hyponymy relations are generated by different types of variations, some of which can be considered as more reliable than others, it is possible to distinguish among classes of hyponymy, according to the degree of confidence. For example, in the figure, the information that a *"cockpit"* is an *"enclosure"* or a *"compartment"* is derived from WordNet, and as such it is considered less reliable than the information obtained from the set of domain descriptions derived from the background document collection.[7]

It is clear that only some aspects of domain knowledge can be captured with the procedures described in this chapter. More advanced ways of capturing domain knowledge are the object of our current research and are briefly described in Chapter 9, "Relation Mining over Biomedical Literature". In any case, our goal is to make use of such knowledge structure in the question answering process. In the next chapter we will show how we did this in case of an existing restricted-domain question answering system.

[7] In the rest of our work, we did not make further use of the distinction between different 'degrees' of the hyponymy relation, and we considered all of them as equivalent.

Chapter 6

Domain descriptions in the QA process

As Question Answering involves a high degree of linguistic processing, domain descriptions quickly become a major obstacle to computational efficiency. To produce a syntactic representation for each sentence, the descriptions need to be identified as phrasal units. As only the word *compartment* in the description *overhead stowage compartment* interacts with other sentence words, *overhead stowage* can be effectively ignored to produce the sentence parse. This is an advisable strategy as otherwise the parser might attempt to combine *overhead stowage* with the wrong sentence words producing multiple possible analysis for a given sentence, leaving the correct parse still to be chosen. Less importantly perhaps, but still significantly, the parser also wastes effort in assigning an internal structure to the description itself. The internal syntactic structure of descriptions is notoriously difficult to identify correctly. A parser might either fail to find the correct structure, or might deliver multiple possible structures, among which a disambiguation is nearly impossible.

Often, computational approaches to the analysis of technical text simply ig-

nore the problem, and attempt to parse with a unified grammatical framework both the 'plain language' and the 'technical language' in the documents. However, ignoring the collocational nature of domain descriptions not only leads to an increase in the complexity of parsing; very often it also leads to the generation of a number of spurious syntactic interpretations, which in turn require more complex disambiguation procedures.

In the research reported here, we took instead an opposite approach and proposed to treat all domain descriptions separately, by first recognizing them (as described in Chapter 4, "Extraction of Domain Descriptions") and their potential variants (as described in Chapter 5, "Structuring of domain descriptions"), and later using them as 'fixed units' in further processing. In this chapter we show how such an approach was successfully applied within the context of a question answering system, by extending and modifying ExtrAns (Section 2.8), a system originally developed to answer questions about the unix man pages.

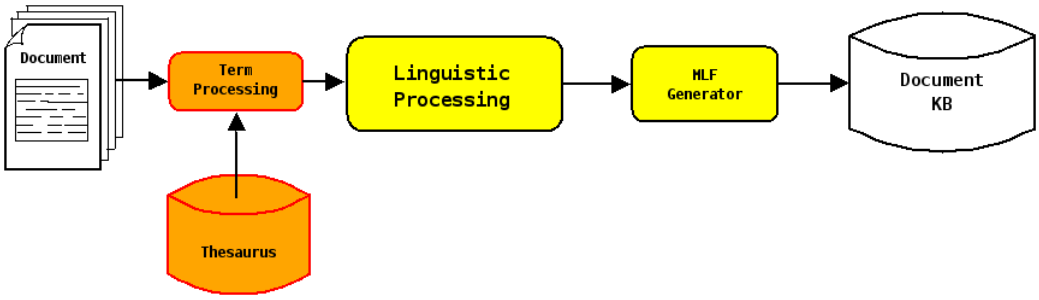
6.1 Syntactic Analysis

In order to deal effectively with the domain descriptions, the original QA system had to be modified on many levels. We therefore describe again the main workings of the system, focusing here on those aspects that had to be modified.

The syntactic analysis begins with the tokenizer. Sentences are split into the units of analysis which optimize processing - words and sentence boundaries are all identified. Domain descriptions, recognized in a separate phase (Chapter 4, "Extraction of Domain Descriptions") are now treated using a dedicated approach explained below.

An efficient lookup procedure identifies in the running text the domain de-

Figure 6.1 Offline processing (extensions in orange, modifications in yellow)



scriptions (and their variants) which have been previously stored in the system's computational thesaurus (see Section 5.3 and Figure 6.1). As the head of a multi-word description controls sentence level syntactic behaviour, each description is considered as a single unit and assigned the syntactic requirements of the head. As such, they are identified as either singular (`DESCRIPTION.s`) or plural (`DESCRIPTION.p`) noun phrases. In parallel to the assignment of syntactic features, a semantic value is assigned to the description, which corresponds to the identifier of the synset to which it belongs.

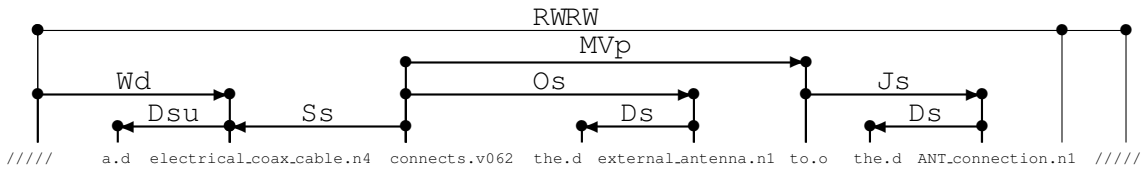
In this way the same description (or descriptions belonging to the same synset) are treated syntactically as noun phrases (either singular or plural).[1] At the semantic level, descriptions belonging to the same synset are equated, all being replaced by their synset identifier. It could be argued that the semantic representation of singular and plural nouns should be different, however in our application we deliberately choose to ignore a number of semantic differences that have no impact on the problem we aim at solving.

A possible alternative approach would be to use the internal structure of descriptions (detected as described in Section 5.2), in the process of building

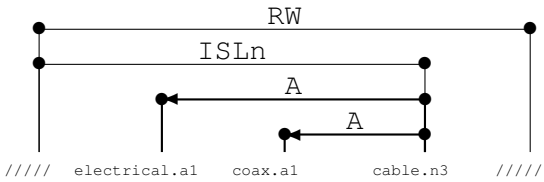
[1]Generally speaking, a description does not necessarily have to be a noun phrase, though they always are in our domain.

the semantic representation of the sentences. This would however require mantaining a dual representation for each domain description at various levels of processing, once as a frozen syntactic unit (useful for parsing) and once as a compound, where the head carries the syntactic information. At present, we find such an approach to be cumbersome while the solution that we have adopted provides for a neater flow of information. We do not rule out however the possibility of exploiting the internal structure of descriptions at a later stage in our research.

Figure 6.2 Examples of LG output



(a)



(b)

Parsing is based upon the robust, dependency-based Link Grammar (LG) parser [Sleator and Temperley, 1993], which is able to handle a wide range of syntactic structures [Sutcliffe and McElligott, 1996b]. LG uses linkages to describe the syntactic structure of a sentence. Each word carries linking requirements (singular determiners 'look for' singular nouns etc.), a linkage representation of a sentence (Figure 6.2a) satisfies all of these individual requirements in a connected graph without any cross-over links. Links con-

nect pairs of words in such a way that the requirements of each word described in the sentences are satisfied, the links do not cross, and the words form a connected graph. An ability to predict the syntactic requirements of 'unknown' words and to process ungrammatical sentences by optionally ignoring some tokens, ensures that an analysis of each sentence is returned. This is vital in the construction of the semantic representation.

In more detail, in the example in Figure 6.2a, the link `Wd` connects the subject *coax_cable* to the wall.[2] The wall functions as a dummy word at the beginning of every sentence and has linking requirements like any other word. `Ss` links the transitive verb *connects* with the subject on the left, the verbal head on the right. The transitive verb and its direct object *external_antenna*, that acts as the head of a noun phrase, are connected by the `Os` link. `MVp` connects the verb to the modifying prepositional phrase. Finally, the link `Js` connects the preposition *to* with its object *ANT_connection*.

Processing the tokens inside multi-word descriptions individually would introduce additional linking requirements. In the best case, modifiers are all connected to the head (Figure 6.2b), identifying the descriptions as a phrasal unit but offering only a superficial representation of the internal structure. In more complex sentences, such modifiers might also wrongly link to words outside the description, resulting in multiple parses for the given sentence. The single token approach that we have adopted requires only that the linking properties for tokens of the types `DESCRIPTION.s` and `DESCRIPTION.p` be added to the LG lexicon.

Exploiting the atomicity of the domain descriptions previously identified during pre-processing blocks the possibility of erroneous parses, and also saves the computational expense needed to disambiguate between the alternatives. Furthermore, the risk of a parse which involves only fragments

[2]The wall is an artificial constituent introduced by LG as the 'root' of the analysis.

of a domain description (which should be treated as an indivisible unit) is avoided. Experimental results [Rinaldi et al., 2002a, Rinaldi et al., 2003b] show that, using this approach, the number of possible parses is reduced in average by almost 50%.

Thus, reducing the complexity of the material to be parsed by treating multi-word descriptions as atomic elements reduces both the space and time requirements for the parsing process, and can have a dramatic impact on the automatic processing of technical documentation, as these results apply to all domains and texts with a high frequency of domain-specific descriptions.

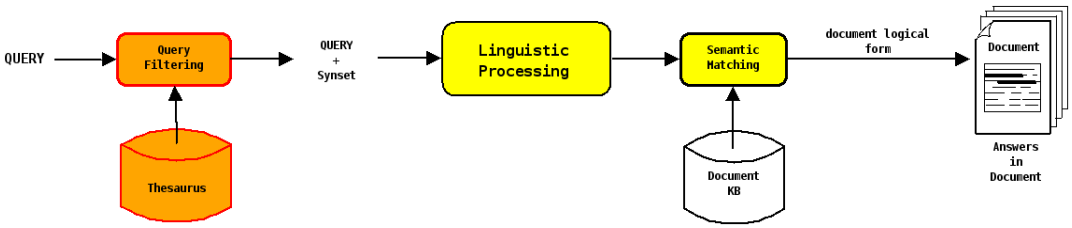
The additional effort required for the analysis of the internal structure of the descriptions might be worthwhile if an accurate internal representation of their structure was possible. However, any parser with a sufficiently rich grammar would deliver a number of potential structures, among which disambiguation is extremely difficult. For example, a typical structure assigned by Link Grammar to a domain description is shown in (Figure 6.2b): additional modifiers add the link `A` (adjectival modifier) or the link `AN` (nominal modifier) to the head of the phrase. Whilst this structure may correctly describe some descriptions (*underfuselage off-centered door*), arbitrary application to *air conditioning system*, *electrical coax cable* or the extension to *no smoking/fasten seat belt (ns/fsb) signs* fails to capture the more subtle patterns of modification.

In a more traditional parsing approach, a clear distinction is drawn between the grammar, the lexicon and the parsing algorithm. In this case, either the grammar does not have sufficient coverage, and therefore some of the possible structures are missing (for example, a few grammars would not cater for the analysis "[[underfuselage] [off-centered] door]"), or it overgenerates, leaving the disambiguation problem open. Link Grammar presents the additional difficulty that it conflates grammar and lexicon. As all the

grammatical information is coded within lexical entries, it is problematic to provide a general fix for the problem of missing analysis. This is one of the reasons why, in our more recent research (see Chapter 8, "A QA application for biomedical literature" and Chapter 9, "Relation Mining over Biomedical Literature"), we are moving away from Link Grammar.

6.2 Semantic Analysis

Figure 6.3 Online processing (extensions in orange, modifications in yellow)



At the semantic level, the main innovation is the replacement of all descriptions and their variants with their synset number. This happens in the offline stage while processing the documents, and in the online stage while processing the queries (see Figure 6.1 and Figure 6.3). This approach leads to a degree of normalization which reduces the (multiple) descriptions referring to a domain concept to a unique identifier. In the application this removes the need for a query description and a document description to be identical: variations in domain descriptions between query and document no longer prohibit the logical proof criteria, if they belong to the same synset.

Directed dependency relations are used to express verb-argument relations, as well as modifier and adjunct relations. This information is converted into the Minimal Logical Form (MLF) representation, which encodes the

fundamental meaning of sentences.

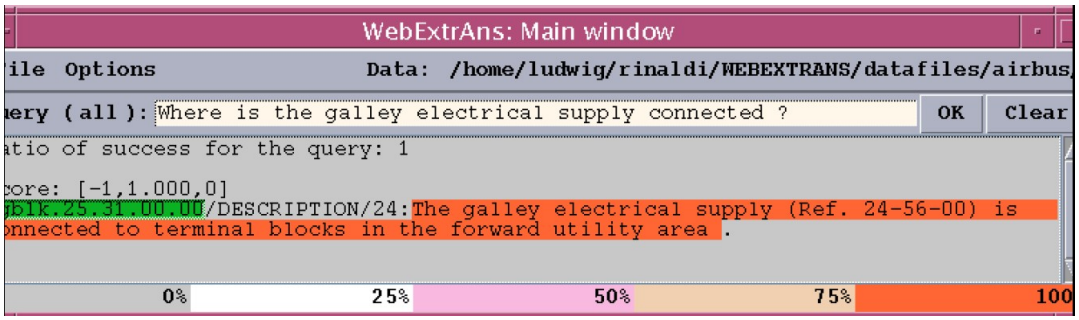
As we have seen in Section 2.8, the MLFs are expressed as conjunctions of predicates with all the variables existentially bound with wide scope. For example, the MLF of the sentence *"A coax cable connects the external antenna to the ANT connection"* is:

```
(1) holds([1]),
    object(s_coax_cable, o2,[v3]),
    object(s_external_antenna,o3,[v4]),
    object(s_ANT_connection,o4,[v5]),
    evt(connect,[1],[v3,v4]),
    prop(to,p1,[[1],v5]),
```

Three multi-word descriptions are contained in this sentence. On the basis of their previous identification during the preprocessing stage (Chapter 4, "Extraction of Domain Descriptions"), the system is capable of recognizing them and translating them into single predicates. In the logical form above they are represented as: `v3`, a *coax cable*, `v4` an *external antenna* and `v5` an *ANT connection*. Notice that if a different description was used for the same object (e.g. *"coaxial cable"* or *"externally located antenna"*), the logical form would remain identical. The embedded predicate names `s_coax_cable` and `s_external_antenna` apply to all descriptions within the same synset. The entity [1] represents the 'connect' event involving two arguments, the *coax_cable* and the *external_antenna*. This reified event, [1] , is used again in the final clause to assert that the event happens '*to*' `v5` (the *ANT_connection*). This is the utility of reification: yielding the additional arguments o2, o3, o4 and [1] as hooks for additional modifiers to be attached to the entities they denote.

6.3 Answer Extraction

Figure 6.4 . An example of output - query window



As discussed, the documents are processed in an offline stage and translated into MLFs, which are stored in a Knowledge Base.

Answer extraction is performed by analysing the queries in an online stage, using exactly the same techniques, and then finding those sentences whose MLFs form a superset of the MLFs of the question. To make this happen, the MLFs are translated into Prolog predicates and Prolog's theorem prover is used to find the answers.

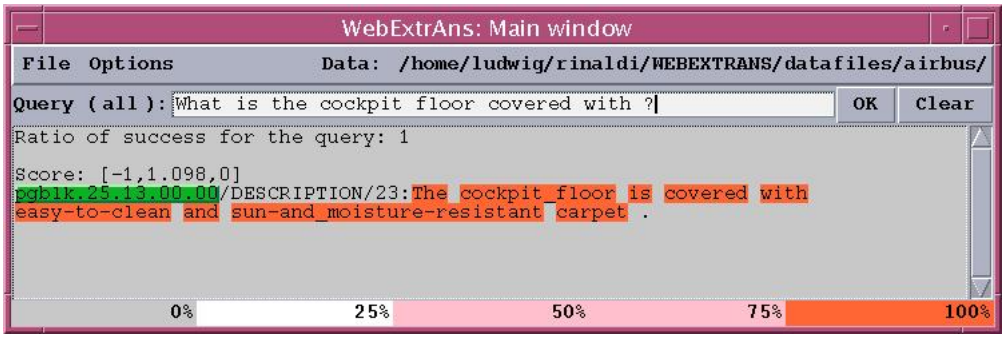
The logical form of the question *"How is the external antenna connected ?"* is:

```
(2) holds(v1),
    object(s_external_antenna,o2,[v5]),
    evt(connect,v1,[v4,v5]),
    object(anonymous_object,v3,[v4]).
```

The variables introduced in a question MLF are converted into Prolog variables. The resulting MLF can be run as a Prolog query that will succeed provided that there has been an assertion in the text that the *external antenna*

is connected to or by *something*. This *something* is the anonymous object of the query.

Figure 6.5 Example of QA session



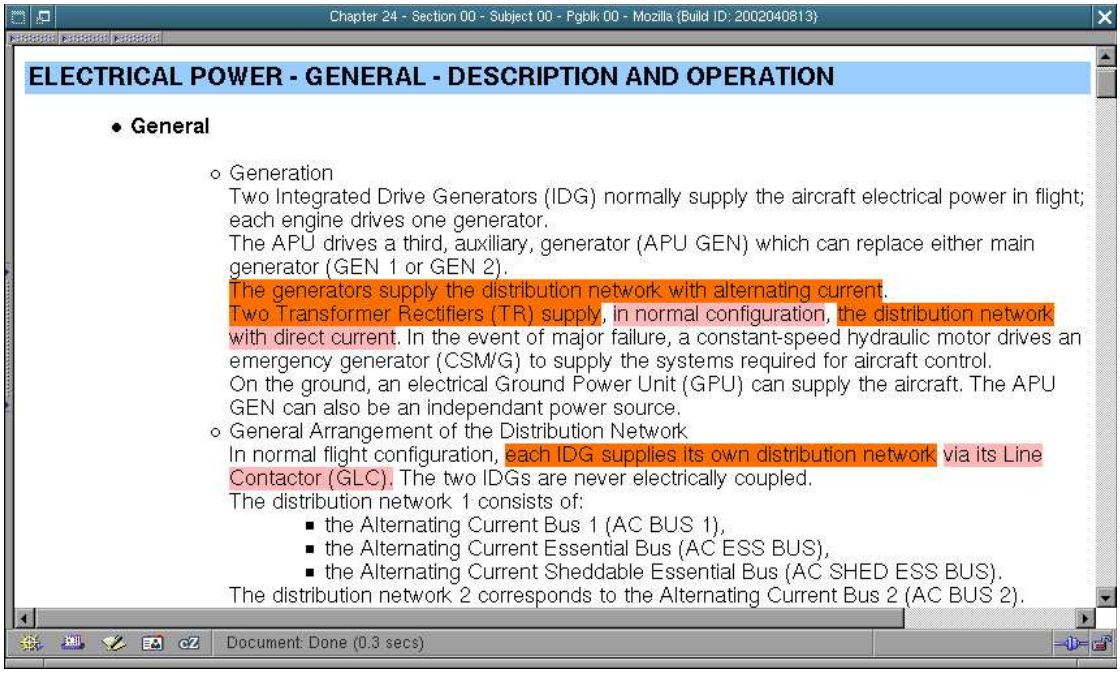
A sentence identifier and a pointer (indicating the tokens from which the predicate has been derived) are attached to each predicate of a MLF in the knowledge base. This information matches against additional variables attached to the predicates in the question (not shown in the example above) and is eventually used to highlight those words in the retrieved sentence that contribute most to that particular answer. Examples of the output can be seen in Figure 6.4 and Figure 6.5. When the user clicks on one of the answers provided, the corresponding document is displayed with the relevant passages highlighted (see Figure 6.6). If there are no answers or too few answers, the system relaxes the proof criteria as described in Section 6.4 below.

Given that the MLFs are simplified logical forms converted into flat structures, the system may find sentences that are not exact answers but are still related to the user's question. Thus, given the question above, the system may also find sentences such as:

1. *"The external antenna must not be directly connected to the control panel."*

2. *"Do not connect the external antenna before it is grounded."*

Figure 6.6 Answer to the query "How is the distribution network supplied?" - document window



3. *"The external antenna is connected, with a coax cable, to the ANT connection on the ELT transmitter."*

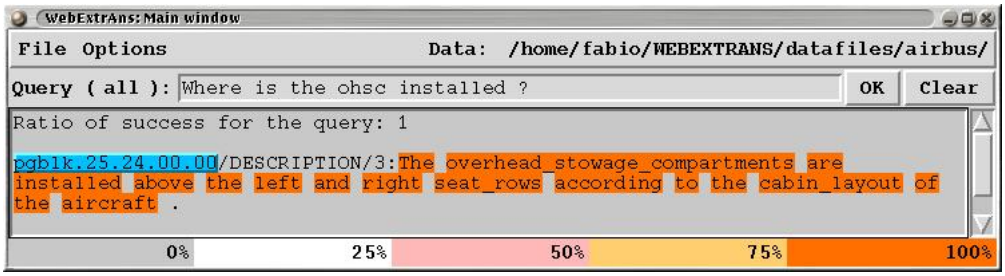
4. *"To connect the external antenna use a coax cable."*

In many real situations, the information provided by the answers above, although not direct answers to the question, might prove to be useful to the user, in particular if a direct answer could not be found.

An additional advantage of the MLFs is that they can be produced without domain knowledge. This makes our technology easily portable to different domains. The only true impact of the domain is during the preprocessing stage of the input text and during the creation of a terminological knowledge base that reflects the specific descriptions used in the chosen domain, their lexical relations and their word senses.

6.4 Extended Search

Figure 6.7 . Query 1: 'ohsc' is a synonym of 'overhead stowage compartment'



Example 6.4.1

```
holds(v_e5)~[], object(s_stowage_compartments,v_o_a1,[v_x4])~[4],
evt(install, v_e5,[v_a2,v_x4])~[1,2,3,4,5],
object(anonym_object,v_o_a3,[v_a2])~[]
```

Example 6.4.2

```
object(s_stowage_compartment,A,[B]),
evt(install,C,[D,B]), object(E,F,[D|G])
```

In the extended search mode the hyperonymy relations, detected as described in Section 5.2, can be put to good use. While in normal search mode only the synonymy relation is taken into account,[3] in extended search mode the logical form of the query is enriched with hyponyms and hyperonyms of the descriptions that it contains.[4]

For example, the logical form (Example 6.4.1) of the query *"Where are the stowage compartments installed?"* is initially translated into the Horn query

[3] Figure 6.7 is an example of query which locates a valid answer by using the synonymy information.

[4] Actually, only 1 level of hyperonymy (above the current description) is considered. It is in fact clear that an arbitrary recursion up the levels of the taxonomy would lead to a too broad query.

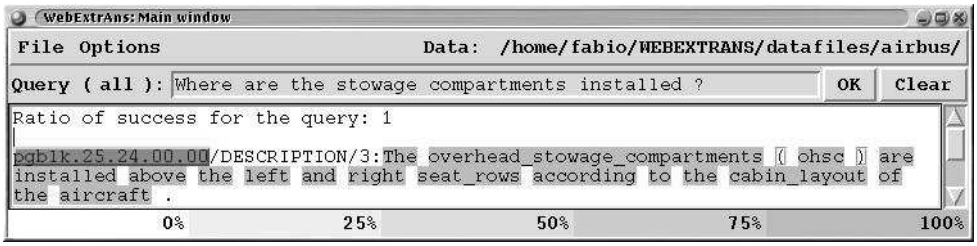
(Example 6.4.2). This means that an object which is a stowage_compartment is involved in a install event with an anonymous object. If there is a MLF from the document that can identify the anonymous object (i.e. where the install event is) the answer is found. If not an expansion of the Horn query to include all hyponymy and hyperonymy possibilities is tried (see Example 6.4.3).

Now the alternative objects are in a logical `OR` relation. This Horn query finds the answer in Figure 6.8.

Example 6.4.3

```
(object(s_stowage_compartment,A,[B]);
object(s_overhead_stowage_compartment,A,[B])),
evt(install,C,[D,B]), object(E,F,[D|G])
```

Figure 6.8 overhead_stowage_compartment is an hyponym of stowage_compartment



6.5 Comments

A potential drawback of this search strategy is the strong reliance on the automated synonymy identification procedure. Because some degree of error is unavoidable, we might end up with 'ambiguous' synsets, containing descriptions that are not necessarily synonyms. In our application, this degree of error has proven to be negligible, and thus acceptable. In a situation

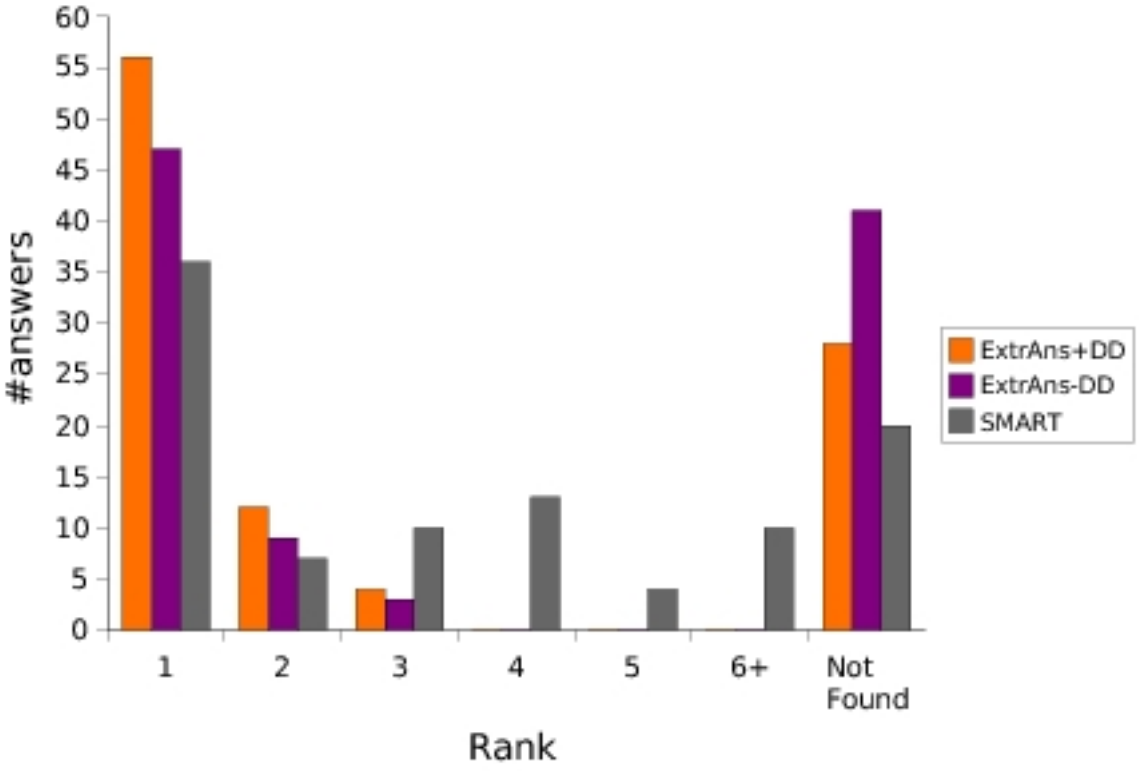
where such minimal error is not acceptable, a manual verification of the synonymy and hyponymy relations would be required.
Further, it is the very notion of synonymy that could be put into question, as it might happen that variants obtained with a series of transformational steps end up being very distant from the description that the process started from. Also, it could be argued that detection of synonymy cannot be always stated in boolean terms, i.e. in some cases we could say that two words or descriptions are synonyms to some degree.[5] However this latter problem, which is widespread in general language, is less relevant in technical domains, where the intended referent of a description is (in general) very precisely identifiable.
The level of ambiguous synsets in the AMM thesaurus (1.2%) is acceptable for our precision requirements. However, the definitions of the semantic relations between descriptions (especially synonymy) need to be tested and refined across different techincal domains.

6.6 Evaluation

In order to set up an evaluation framework for our system, we decided to consider an IR system as a baseline, even if the standard measures of *precision* and *recall* are not ideal for an Answer Extraction system. In particular recall is significantly less important than precision, as the aim of such a system is to provide (at least) one correct answer, rather than all the possible answers in a given collection.
In the QA track of TREC a measure that is commonly used to evaluate QA systems is the Mean Reciprocal Rank (MRR). The Rank of a given result is the position of the first correct answer in the output list of the system. Over

[5] Unfortunately not easily measurable.

Figure 6.9 Answers at different ranks



a given set of answers, MRR is computed as the mean of the reciprocals of the ranks for all the answers (see Section 2.3).

The particular evaluation that we present here is targeted at the new application in the AMM domain. We devised 100 questions by selecting interesting passages from the manual and formulating questions to which those passages could be an answer. The questions were submitted to both our system and the selected IR system (SMART). We used for comparison two versions of ExtrAns: first the original version, without special treatment of domain descriptions ("ExtrAns-DD"), then our extended version, with special treatment for domain descriptions ("ExtrAns+DD").

While in general our system retrieves a short number of answers, that can

be easily checked manually, SMART retrieves a ranked list of documents. As manual inspection of all the documents retrieved by SMART would be impossible, we decided to set an arbitrary threshold (at 10), i.e. if no valid answer was contained in the first ten retrieved documents, we classified it as *"Not Found"*.

The diagram (Figure 6.9) shows how many correct answers are found at each rank (1 to 5, answers from 6 to 10 are considered together) for each of the 3 systems used in the evaluation. As it can be seen, both versions of our system find fewer answers than SMART (even ruling out all answers ranked greater than 10). Therefore recall is clearly higher for SMART. However in the majority of cases, when the system does find the answer, it places it in the first position.

There are very few cases where an answer at a lower rank is correct while answers at higher ranks for the same question are not. It does happen that our system retrieves incorrect answers together with the correct one, but in that case the correct one is almost always ranked first.

For the particular evaluation that we have presented, the 'ExtrAns+DD' system obtains a MRR of 0.63, which is a very good result if compared with results obtained in TREC. However we should stress that such a comparison is misleading, as our evaluation is far more restricted than those carried out in TREC. Besides, our system at the moment cannot cope with very large volumes of data as seen in TREC. For comparison, the system without domain descriptions ('ExtrAns-DD') obtains a MRR of 0.53, while SMART has a MRR of 0.45.

In general, this evaluation leads us to conclude that our system can provide far higher precision than a generic IR system, at the price of a lower recall. Recall alone however is not interesting. In our situation, quick location of the precise answer is the most important factor. Relevant documents that

are ranked poorly are likely to remain unnoticed by the user (see Section 2.3 for a fuller discussion of these issues).

As for the role played by domain descriptions, the evaluation clearly shows that, in addition to the benefits in parsing that we have previously described, they also help in retrieving more answers. The ExtrAns+DD system found 13 answers that the ExtrAns-DD system could not find. For example, a question like *"Where is the ohsc installed?"* could not be answered by the system without domain descriptions.

6.7 Conclusion

In this chapter we have shown how an existing restricted-domain question-answering system was ported to a new domain and enhanced with new capabilities, allowing it to deal efficiently and flexibly with domain descriptions.

The successful outcome of this experiment led us to consider the potential application of the developed techniques to another domain, as we will describe in the following chapters.

Chapter 7

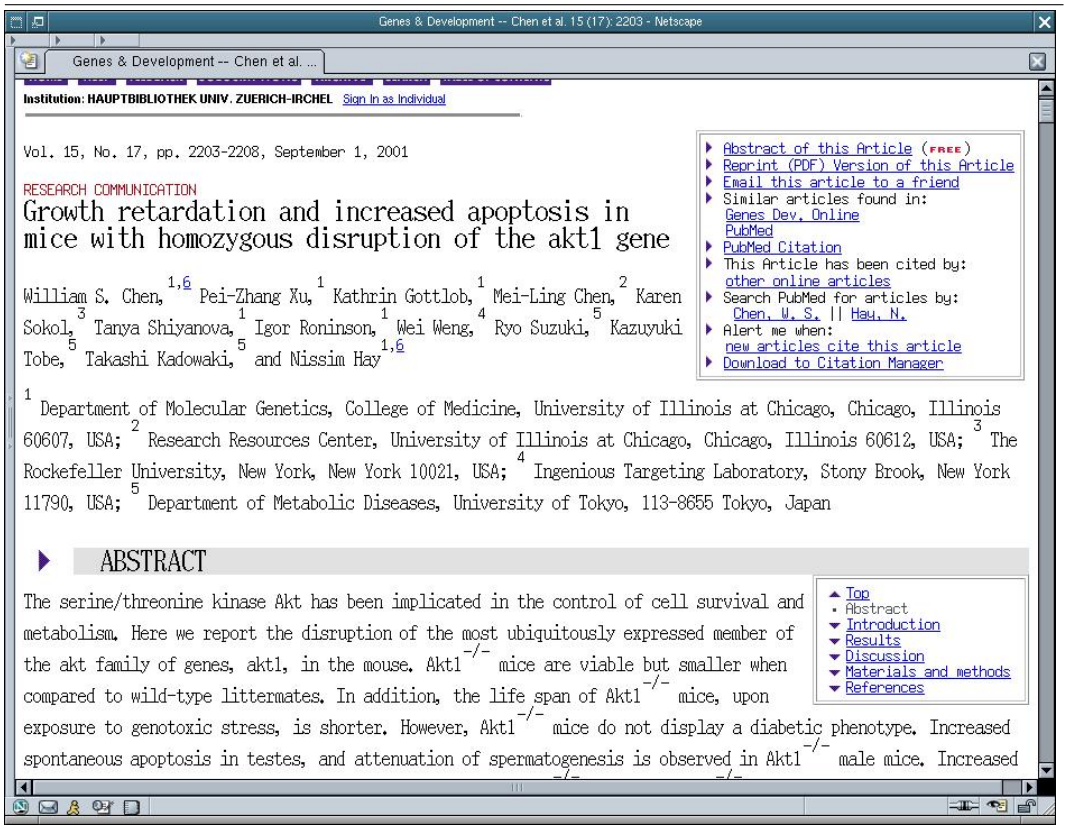
The biomedical domain: motivations and background research

The quantity of research results in the area of molecular biology is growing at such a pace that it is extremely difficult for individual researchers to keep track of them. As such results appear mainly in the form of scientific articles (an example is shown in Figure 7.1), it is necessary to process them in an efficient manner in order to be able to extract the relevant results. Many databases aim at consolidating the newly gained knowledge in a format that is easily accessible and searchable (e.g. UMLS, Swiss-Prot, OMIM, GeneOntology (GO), GenBank, EntrezGene[1]). However the creation of such resources is a labour intensive process: relevant articles have to be selected and accurately read by human experts looking for the needed information.[2] Besides, there can be a significant time lag between the publication of a result and its introduction into such databases.

[1]Formerly known as LocusLink

[2]This process is referred to as 'curation' of the article.

Figure 7.1 Example of document to be analyzed



The various genome sequencing efforts (for different organisms) have resulted in the creation of large databases containing gene sequences. However such information is of little use without the knowledge of the function of each gene and its role in biological pathways. Understanding the relationships within and between these groups is central to biology research and drug design as they form an array of intricate and interconnected molecular interaction networks which is the basis of normal development and the sustenance of health.

One of the core problems in exploiting scientific papers in research and clinical settings is that the knowledge that they contain is not easily accessible [Stapley and Benoit, 2000]. Therefore, there is a considerable activity

in the area of automatic extraction of biological information from scientific literature. Specific initiatives that focus on entity extraction are numerous. Interest in more challenging types of activities (e.g. extraction of relations or complex events) is growing. However, resources (Annotated Corpora, Treebanks, etc.) are needed in order to bootstrap the development of the advanced tools of tomorrow.

There are precious few examples of question answering over genomics literature. Question answering in biomedicine is surveyed in detail in [Zweigenbaum, 2003], in particular regarding clinical questions. An example of a system applied to such questions is presented in [Niu et al., 2003], where it is applied in a setting for evidence-based medicine. This system identifies specific 'roles' within the document sentences and the questions; determining the answers is then a matter of comparing the roles in each. To achieve this, natural language questions are translated into the PICO format [Sackett et al., 2000], which is essentially a template of the roles contained in the question. The identification of roles requires additionally hand-written rules which are time consuming to produce and domain specific.

The relevance of question answering for the genomics domains is testified by the "roadmap" for the genomics track of TREC [TREC-GE, 2003] which foresees question answering as an advanced topic.

7.1 Related Work

7.1.1 Entity Extraction for Biomedical Literature

Entity Extraction is considered the fundamental task in information extraction (IE). Classical IE, as defined in the MUC Conferences [Chinchor, 1998b], is concerned with a limited set of entities (persons, locations, organizations).

Applications in the Bio domain need a different set of entities (genes, proteins, drugs, cell types, pathways, etc.). Another difference between classical IE and IE for biomedical literature is that the tools might not have to to identify and mark up all entities mentioned in documents, it might be sufficient to deliver a list of the entities mentioned in the text. This is the approach taken in Task 1 of the BioCreAtIvE [3] competition (for details see Section 7.1.3): tools are asked to retrieve the list of genes mentioned in a document, and map each of them to its unique name or symbol.

While named entity recognition systems over news text is now reaching performance levels near to that of human annotators, named entity recognition in the biomedical domain is particularly difficult, since there are ambiguous and synonymous entities, few standardizations, lack of explicit marking such as capitalization, nick-naming, number-incorporation, (sometimes ad hoc) abbreviations and possible word permutations in multi-word descriptions. Also, it is impossible to come up with a complete list of entities as the field is developing too fast. Rule-based as well as machine learning or hybrid named entity identifiers are known. They use dictionaries such as Swiss-Prot and/or fuzzy letter-sequences. There are many successful systems (for example [Franzen et al., 2002, Hanisch et al., 2003, Koike and Takagi, 2004]).

There are also several free tools which focus on entity extraction in the genomics domain. Those tools are generally statistical, based on various machine learning methods. Some examples are the following:

- YAGI[4] is a command-line annotation tool that uses conditional random fields (CRFs) trained on the BioCreAtIvE Task 1a dataset to identify gene names in biomedical text. It annotates only genes and gene

[3] <http://www.mitre.org/public/biocreative>

[4] <http://www.cs.wisc.edu/~bsettles/yagi>

products (such as a protein and RNA).

- LingPipe[5] is a suite of Java tools designed to perform linguistic analysis on natural language data. Its named entity extraction component is trained for both English news and English genomics domain.
- AbGene[6] is trained on Medline abstracts
- GAPSCORE[7] identifies the names of genes and proteins. It can be used via XML-RPC.

[Spasić et al., 2003] uses domain-relevant verbs to improve on terminology extraction. The co-occurrence in sentences of selected verbs and candidate terms reinforces their termhood. [Stapley and Benoit, 2000] measure statistical gene name co-occurrence and graphically displays the results so that an expert can investigate the dominant patterns. The PubMed[8] system uses the Metathesaurus[9] as a controlled vocabulary to index biomedical abstracts. This allows efficient retrieval of abstracts from medical journals, making use of hyponymy and lexical synonymy to organize the terms. However, [Cimino, 2001] criticizes the UMLS because of the inconsistencies and subjective bias imposed on the relations by the manual creation of the relationships among its concepts.

Still, the vast majority of research in this area is founded upon utilizing the UMLS MetaThesaurus. Whilst this is fine for research purposes, the time lag

[5] <http://www.alias-i.com/lingpipe/>

[6] <ftp://ftp.ncbi.nlm.nih.gov/pub/tanabe/AbGene/>

[7] <http://bionlp.stanford.edu/gapscore/>

[8] <http://www.ncbi.nlm.nih.gov/pubmed/>

[9] The Metathesaurus is a very large, multi-purpose, and multi-lingual vocabulary database that contains information about biomedical and health related concepts, their various names, and the relationships among them. It is a component of the UMLS (Unified Medical Language System).

between identifying novel genes and pathways and including them in the UMLS tools negates any competitive edge in discovering relations. By the time they are included in the UMLS they are 'old news' [Park et al., 2003].

7.1.2 Relation Extraction for Biomedical Literature

7.1.2.1 Relation Extraction using Patterns

While a majority of the applications of natural language processing techniques in the domain of molecular biology tend to focus on entity discovery, such as genes and proteins (see for instance [Johnson, 2002b] and [Ananiadou and Tsujii, 2003]) there are some significant efforts aimed at detecting relations among those entities.

[Craven and Kumlien, 1999] identifies possible drug interaction relations (predicates) between names of proteins and chemicals using a 'bag of words' approach applied to the sentence level.

[Ono et al., 2001] reports on extraction of protein-protein interactions based on a combination of syntactic patterns. The authors employ a simple dictionary lookup procedure to identify protein names in the documents to analyze, then select sentences that contain at least two protein names, which are then parsed with very simple part-of-speech matching rules. The rules are triggered by a set of (stemmed) keywords which are frequently used to name protein interactions (e.g. 'associate', 'bind', etc.) and can identify negative statements (again by matching specific words, such as 'not').

[Sekimizu et al., 1998] uses frequently occurring predicates and identifies the subject and object arguments in the predication. In contrast [Rindflesch et al., 2000] uses named entity recognition techniques to identify drugs and genes, then identifies the predicates which connect them. [Cimino and Barnet, 1993] uses 'if then' rules to extract semantic relationships between the

medical entities depending on which MeSH headings these entities appear under. For example, if a citation has *"Electrocardiography"* with the subheading *"Methods"* and has *"Myocardial Infarction"* with the subheading *"Diagnosis"* then *"Electrocardiography"* diagnoses *"Myocardial Infarction"*.

7.1.2.2 Relation Extraction using Parsing

In the BioMedical domain, the opinion that there is not yet a viable alternative to shallow parsing has been predominant [Shatkay and Feldman, 2003]. Since the approaches of the authors they quote, large-scale parsing has made tremendous progress, however, both in terms of speed and accuracy, so that their verdict merits re-assessment. Recent developments in, for example, question answering have shown that deep-linguistic processing is beneficial (see Chapter 2, "Overview of Question Answering Systems").
Currently, few NLP approaches in the BioMedical domain include full parsing. In the following we summarize all the research projects (of which we are aware) that include full parsing – or are getting close to it – for the BioMedical domain.

[Park et al., 2001] presents experiments on parsing MEDLINE abstracts with Combinatory Categorial Grammar (CCG). Compared to state-of-the-art parsing speed, the system is too slow for practical application (13 minutes for 200 sentences). A small evaluation on 492 sentences yields 80% precision but only 48% recall in extracting domain-specific relations.

[Hahn et al., 2002] describes a medical IE system that uses a dependency grammar [Bröker et al., 1994]. Only German versions of the parser are described. An evaluation with promising results is reported, but only on three low-level relations: auxiliaries, genitives and prepositional phrases.
An early work which uses full parsing for biomedical literature is [Yakushiji et al., 2001]. The authors belong to the research group that has made the

GENIA corpus available, and are currently building the GENIA treebank (which is based on the partially corrected output of their parser). They use a widely established formal grammar, HPSG, and they have expertise in robust parsing; [Miyao et al., 2004] is probably the first HPSG parsing approach that scales up to the entire Penn Treebank. [Yakushiji et al., 2004] use the approach to find a selected number of verbs in medical corpora.

[Friedman et al., 2001] describes a system (*GENIES*) which extracts and structures information about cellular pathways from the biological literature. The system relies on a term tagger using rules and external knowledge. The terms are combined in relations using a syntactic grammar and semantic constraints. It attempts to obtain a full parse to achieve high precision, but often backs off to partial parsing to improve recall. It groups 125 selected anchor verbs into 14 semantic classes, and it even includes some nominalisations. They report an impressive precision of 96% and 63% recall, however their evaluation is based on a single journal article.

The *PASTA* system [Gaizauskas et al., 2003] uses a MUC-inspired template-based information extraction approach. They focus on the roles of specific amino acid residues in protein molecules. The system internally uses a parse-based approach, with a predicate argument representation, but the results that it delivers are used to fill predefined templates. On the basis of this representation they also build a domain model which allows inferences based on multiple sentences. PASTA is perhaps the only parsing-based BioNLP system that has been given an extensive and thorough evaluation. Using the MUC-7 scoring system on the hard task of template recognition they report 65% precision and 68% recall, which compares well with MUC results.

[Pustejovsky et al., 2002] processes Medline articles (only titles and abstracts) focusing on relation identification. An advantage of their system is

the anaphora resolution module, which can resolve many cases of pronominal anaphora and anaphora of the sortal type (e.g. *"the protein"*) including multiple antecedents (e.g. *"both enzymes"*). Their evaluation is based on the `inhibit` relation. They do not use full parsing, but a finite-state cascade approach in which for example many PPs remain unattached. Their shallow parsing is closer to a full parse than most other systems because they include a subordinate clause level, sentential coordination and a flexible relation identification module.

[Novichkova et al., 2003, Daraselia et al., 2004b] report on their system (MedScan), which involves full parsing. [Novichkova et al., 2003] contains a true broad-coverage evaluation of the coverage of their syntax module, which was tested on 4.6 million sentences from PubMed. Only 1.56 million sentences of these yield a parse, which is 34% coverage. [Daraselia et al., 2004a] reports an impressive 91% precision but only 21% recall when extracting human protein interactions from MEDLINE using MedScan. A main reason for the relatively low recall is because *"the coverage of MedScan grammar is about 51%, which means that information is extracted from only about half of the sentences"* [ibid.].

A comparative analysis of Link Grammar (LG) by [Mollá and Hutchinson, 2003] in 2003 has shown that the precision of Link Grammar [Sleator and Temperley, 1991] was, by then, considerably below state-of-the-art. [Sampo et al., 2004], in a formal evaluation of parsing BioMedical texts with the Link Grammar Parser report an overall dependency recall of 73.1%.[10] Additionally, the syntactic analysis performed by LG is based on unusual syntactic

[10]Notice that the evaluation results mentioned in this section are those reported by the authors, and are not always comparable. For example, the previously reported result on MedScan refers to the task of detecting protein interactions, while the recall mentioned for LG refers to a plain syntactic task.

structures, which limits its usability. On the basis of such evaluations, we concluded that LG, although it had been extremely useful in the original ExtrAns system, was no longer state-of-the-art and had to be replaced by a more recent parsing approach.

7.1.3 Collaborative Evaluation Challenges

Thorough evaluation of Text Mining systems in Biology has only just begun. The earliest of these, the KDD cup [KDD, 2002][11] was based on the information curation process within the Flybase consortium. Three tasks were defined:

- identify publications that are relevant to the curation process
- specify which of the genes mentioned in the paper had their expression actually discussed in it
- distinguish for each expressed gene whether the reported expression product is a transcript or a protein

Since 2003, TREC has been organizing a Genomics track [TREC-GE, 2003], which is organized as two major tasks: Ad Hoc Retrieval and Categorization. The former is plain traditional IR applied to Genomics literature, the latter focuses on classifying documents containing "experimental evidence" allowing assignment of Gene Ontology (GO) codes.[12]
The categorization task is further divided into two subtasks: triage and annotation. The triage task is a classical document categorization task, specialized to this domain. The annotation task involves using GO codes to annotate a given document, which are then compared with manually selected codes.

[11] <http://www.biostat.wisc.edu/~craven/kddcup/>
[12] <http://www.geneontology.org/>

BioCreAtIvE (Critical Assessment of Information Extraction systems in Biology) defines two "biologically meaningful" tasks. The first task requires the participants to find mentions of biological entities (e.g. genes, proteins, etc.) in the given document collection. The second task is of greater complexity, aiming at automatic functional annotation of proteins with respect to the GO classes. It is organized into a number of subtasks:

- find evidence in the document for the GO annotation
- create GO annotations for human proteins
- select relevant papers, which support a GO annotation

In order to make sure that the results come from an automated system (and not human annotators), BioCreAtIvE allows only a limited time between data delivery and return of the results. Further, the volume of data is large enough to make human annotation near impossible. The systems have also to return a phrase (or short extract) from the text that justifies the selection of the annotation, which is then manually checked by Swiss-Prot annotators for correctness.

Chapter 8

A QA application for biomedical literature

In 2004, after the successful QA applications described in Chapter 6, "Domain descriptions in the QA process", we started looking for a new domain which would allow us to put to test the technologies developed so far and explore new technical challenges. The domain of biomedical literature appeared to be particularly attractive, because of the huge quantity of biomedical literature available, the relatively limited (up to that point) application of computational linguistic techniques to the automatic processing of biomedical articles, and the practical interests of institutions and research companies in improving access to the scientific results as described in the literature.

In this chapter we describe (part of) our research activities over the domain of biomedical literature. We started with the goal of applying the techniques for the management of domain descriptions described in the previous chapters to the very specific entities that characterize this domain (e.g. proteins, genes, diseases). We initially aimed at a further application of the ExtrAns system, but we stopped at the level of a demonstrator. However, in the

course of this research, we came to identify an intermediate target (relation mining), which subsequently became the major focus of our activity. In this chapter we describe how we adapted ExtrAns to a collection of biomedical literature, while in the next chapter we briefly introduce our experiments with relation mining.

8.1 Answering Questions in the Genomics Domain

The survey reported in the previous section motivates the interest in experimenting with the techniques described in Chapters 4, 5 and 6 as applied to the genomics domain.

The first step in adapting our QA system to a new domain is identifying the specific set of documents to be analyzed. We have experimented with two different collections in the genomics domain. The first collection (here called the 'Biovista' corpus) has been generated from Medline using two seed lists of gene and pathway[1] names to extract an initial corpus of research papers (full articles). The second collection is constituted by the GENIA corpus [Kim et al., 2003b][2], which contains 2000 abstracts from Medline (a total of 18546 sentences). The advantage of the latter is that domain-specific terminology is already manually annotated, providing therefore the domain descriptions (such as protein names) that we need for our work. However focusing only on that case would mean disregarding a number of real-world problems, and in particular detection of domain descriptions, which forms a core interest of our research.

[1] Pathways are comples biological processes.

[2] <http://www-tsujii.is.s.u-tokyo.ac.jp/GENIA/>

8.1.1 Dealing with formatting information and zoning

The freely available html2text tool[3] strips the HTML tags whilst preserving the basic document structure of sections, paragraphs and figures. Once non-ASCII characters are dealt with, simplistic processing translates this into zones (e.g. title, sections, paragraphs, bibliography). While this process might appear trivial, its importance should not be underestimated. The initial formatting of the articles provides a wealth of structural information, which is of crucial importance for further processing. An approach which simply strips off the HTML tags would miss out on a potentially rich source of information. In order to preserve such information for further processing, some of the HTML markup is preserved and converted into XML.

First we developed an XML based filtering tool which was used to select zones of the documents that need to be processed in a specific fashion (this processing step is often called 'zoning'). Consider for instance the case of bibliography. The initial structure of the document allows easy identification of each bibliographical item. Isolating the authors, titles and publication information is then trivial (because it follows a regular structure). The name of the authors (together with the HTML cross-references) can then be used to identify the citations within the main body of the paper.[4] If a preliminary zone identification (as described) is not performed, the names of the authors used in the citations would appear as spurious elements within sentences, making their analysis very difficult.

Another common case is that of titles. Normally they are nominal phrases rather than sentences. If the parser was expecting to find a sentence, it would fail. However using the knowledge that a title is being processed, we can modify the configuration of the parser so that it accepts an NP as a

[3] <http://userpage.fu-berlin.de/~mbayer/tools/html2text.html>

[4] If citations take the common form of ['author', 'year'], or similar.

correct parse.

8.1.2 Terminology and Domain Descriptions

As we have discussed in Chapter 3, "Domain Specific Collocations and Technical Terminology", the high frequency of domain descriptions in technical text produces a number of different problems when locating answers. A primary problem is the increased difficulty of parsing text in a technical domain due to domain-specific sublanguage. Various types of multi-word descriptions characterize these domains, in particular referring to specific concepts (e.g. genome sequences, proteins). These multi-word expressions might include lexical items which are either unknown to a generic lexicon (e.g. *"arginine methylation"*) or have a specific meaning unique to this domain. Deverbal adjectives (and nouns) are often mistagged as verbs (e.g. *"mediated activation"*, *"cell killing"*). Abbreviations and acronyms, often complex (e.g. bracketed inside NPs, like *"adenovirus (ad) infection"*) are another common source of inconsistencies. In such cases the parser might fail to identify the compound as a phrase and consequently fail to parse the sentence including such items. Alternatively a parser might attempt to 'guess' their lexical category (in the set of open class categories), leading to an exponential growth of the number of possible syntactic parses and often incorrect decisions. Not only can the internal structure of the compound be multi-way ambiguous, also the boundaries of the compounds are difficult to detect and the parsers may try odd combinations of the tokens belonging to the compounds with neighboring tokens.

We have described in Chapter 4, "Extraction of Domain Descriptions" some approaches that might be taken towards extraction of domain descriptions for a specific domain. The GENIA corpus removes these problems com-

Figure 8.1 Relevant sentences with genes and pathways highlighted

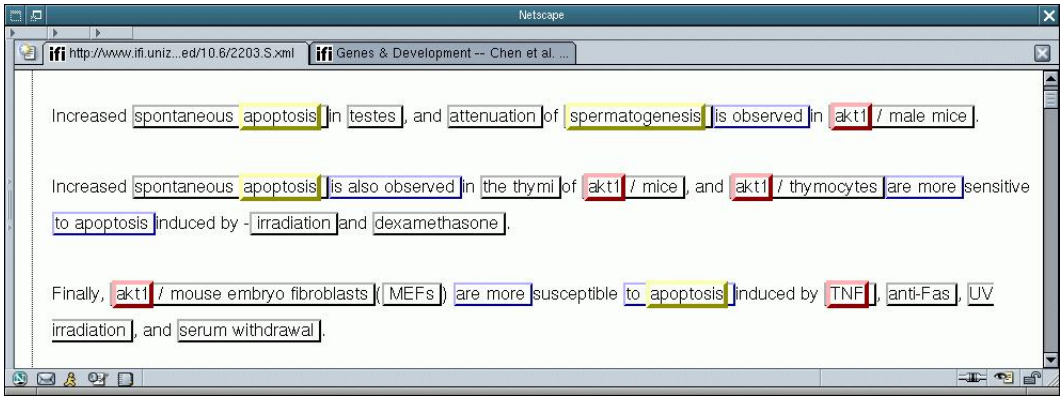
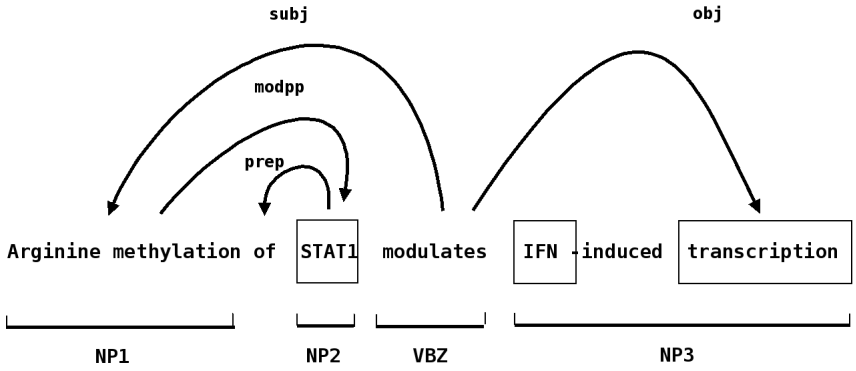


Figure 8.2 An example of dependency-based syntactic analysis.



pletely by providing pre-annotated terminological units, already organized in an ontological structure (the GENIA Ontology). This allows attention to be focused on other challenges of the QA task, rather than getting 'bogged down' with the extraction of the domain descriptions and their organization.

In the case of the Biovista corpus, we had to perform a phase of discovery of the domain descriptions, which was facilitated by the existence of the seed lists of genes and pathways. We first marked up those descriptions which appear in the corpus using additional XML tags. This identified 900 genes and 218 pathways that occur in the corpus (see Figure 8.1).

Next the entire corpus was chunked into nominal and verbal chunks using LT Chunk [Finch and Mikheev, 1997]. Ignoring prepositions and gerunds the chunks are a minimal phrasal group - represented as the square braces in Figure 8.2. The corpus descriptions are then expanded to the boundary of the phrasal chunk they appear in. For example, NP3 in Figure 8.2 contains two units of interest producing the new description "*IFN-induced transcription*". The initial 1118 corpus descriptions were expanded into 6697 new candidate descriptions. 1060 of them involve a pathway in head position and 1154 a gene. The remaining 4483 candidate descriptions involve a novel head with at least one gene or pathway as a modifier.

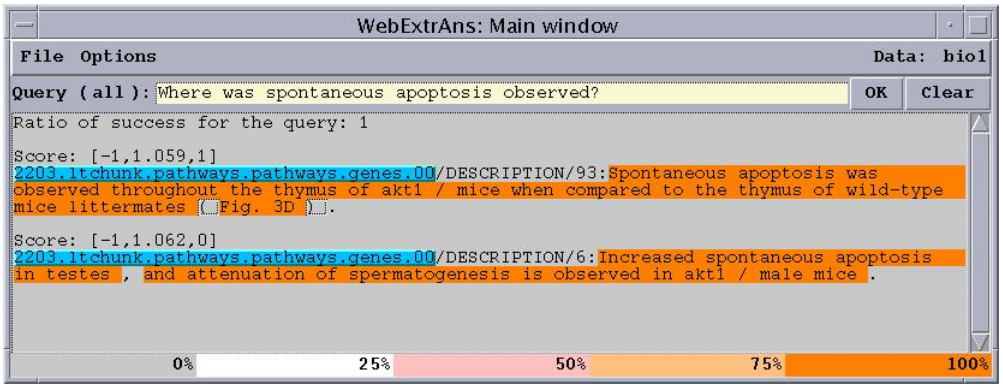
Once the set of descriptions is available, it is necessary to detect their relations in order to exploit them. We have focused our attention in particular to the relations of synonymy and hyponymy, which are detected as described in Chapter 5, "Structuring of domain descriptions" and gathered in a thesaurus.

8.1.3 Parsing

The main innovation at the parsing level is the replacement of the Link Grammar parser with a new statistical broad-coverage parser (Pro3Gres), which is as fast as a probabilistic parsers but more deep-linguistic because it delivers grammatical relation structures which are closer to predicate-argument structures than the linkages delivered by Link Grammar, and therefore can be more easily converted into MLFs. The evaluation reported in [Schneider et al., 2004c] shows that it has state-of-the-art performance.

Figure 8.2 displays the three levels of analysis that are performed on a simple sentence. A process of expansion yields `NP3` as a complete candidate description. However, `NP1` and `NP2` form two distinct, fully expanded noun

Figure 8.3 Example usage of the Bio-QA system



phrase chunks. Their formation into a noun phrase with an embedded prepositional phrase is recovered from the parser's syntactic relations giving the maximally projected noun phrase involving a description: *"Arginine methylation of STAT1"* (or juxtaposed *"STAT1 Arginine methylation"*). Finally, the highest level syntactic relations (`subj` and `obj`) identifies a transitive predicate relation between these two candidate descriptions.

The usage of a deep-linguistic dependency parser partly simplifies the construction of MLF. First, the mapping between labeled dependencies and a surface semantic representation is often more direct than across a complex constituency subtree [Schneider, 2003b], and often more accurate [Johnson, 2002a]. Dedicated labels can directly express complex relations, the lexical participants needed for the construction are more locally available.

An example of interaction with the QA system can be seen in Figure 8.3.

8.2 Conclusion

In this chapter we have discussed our initial attempts at adapting our question answering systems to the biomedical domain. Although the resulting system is no more than a limited prototype, our activities and background

research into this domain have motivated us to seek new paths of exploitation for the techniques of domain modeling that we had previously developed for the AMM manual.

In particular, we have come to realize that a reliable knowledge base is crucial for a satisfactory question answering system, and that the domain knowledge can be, at least in part, automatically derived from documents. At the same time we have become aware of the limitations of the existing QA solution.

In our current research, briefly outlined in the next chapter, we are seeking ways to overcome such limitations, with the long-term goal of creating a novel type of QA system, that can rely upon a rich knowledge base, which however is not manually created, but is instead automatically derived from a sufficiently large collection of documents.

Chapter 9

Relation Mining over Biomedical Literature

In the process of porting our QA system to the biomedical domain we realized that the amount of domain knowledge provided by domain descriptions and their taxonomic relationships is insufficient to offer a satisfactory QA experience. In particular, a large number of relations which exist among domain descriptions cannot be captured by the techniques that we have so far presented, but would instead require a different type of analysis. In this chapter we introduce techniques that we have recently adopted in order to extract other types of domain relations. While pursuing that goal, we have also come to realize that relation mining over biomedical literature is in itself a major research arena, which has since then become the central focus of our activities.

We discuss first in Section 9.1 the role of deep parsing in relation mining, and in particular the benefits of dependency parsing. A precondition for further developing our approach is to be able to rely on a good quality dependency parser, or an annotated dependency corpus. In Section 9.2 we present our work in creating such a corpus, using an internally developed

parser, discussed in Section 9.3. Finally, in Section 9.4 we present an evaluation of domain relations extracted from the dependency corpus.

9.1 Deep Parsing for Relation Mining

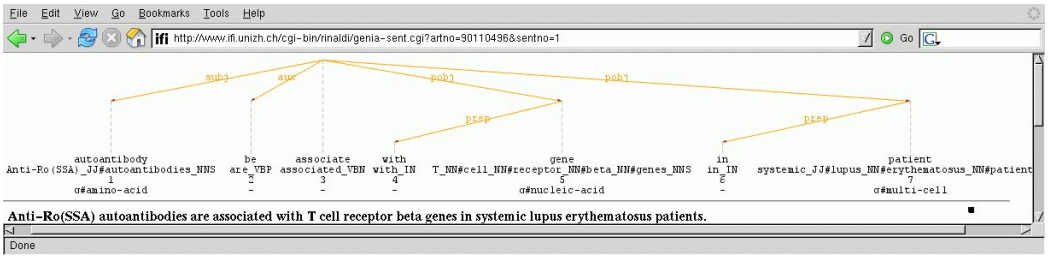
Full parsing, even of complex, highly technical language, is beginning to be possible due to recent developments in parsing technology. Still, as far as we know, few systems exist that show the feasibility of automated relation extraction directly from genomics scientific literature (for details see Section 7.1.2.2). In our research activities, we show that advanced parsing techniques combining statistics and human knowledge of linguistics have matured enough to be successfully applied in real settings.

As an intermediate step towards question answering for the biomedical domain, we aim at developing and refining methods for discovery of interactions between biological entities (genes, proteins, pathways, etc.) from the scientific literature, based on a complete syntactic analysis of the articles, using a novel high-precision parsing approach [Schneider, 2003b]. It has numerous advantages in comparison to Link Grammar: a greater processing speed, fewer erroneous parses, and doesn't go into random mode when it encounters an unkown word.

The GENIA corpus (described in detail in the next section) provides a very interesting test bed for practical experimentation of relation extraction. Its main advantage is that it comes with manually annotated domain descriptions, therefore not requiring a separate phase of extraction. We have used it in further experiments aimed at showing how relationships among domain entities can be extracted in an efficient fashion from a richly annotated corpus [Rinaldi et al., 2006a, Rinaldi et al., 2006b, Rinaldi et al., 2006c].[1] A

[1] A working prototype can be accessed online at <http://www.ontogene.org/>

Figure 9.1 Dependency Structure visualization (via SVG)



similar application has been developed in collaboration with a biomedical company for a different corpus focusing on circadian rythms of Arabidopsis Thaliana [Rinaldi et al., 2007b].[2]

9.2 From GENIA to DEPGENIA

GENIA [Kim et al., 2003b][3] is a corpus of 2000 MedLine abstracts[4] which have been annotated for various biological entities, according to the GENIA Ontology.[5]

In a preliminary phase, we have converted the XML annotations of the GENIA corpus into an equivalent annotation schema based on the a specification developed within the scope of the Parmenides project [Rinaldi et al., 2003a]. There are two main reasons for performing this step. First, in the Parmenides annotation schema all relevant entities are given a unique iden-

[2] Arabidopsis Thaliana is one of the model organisms most frequently used in biomedical research.

[3] <http://www-tsujii.is.s.u-tokyo.ac.jp/GENIA/>

[4] Actually 1999, because article number 97218353 appears twice, curiously with slightly different annotations.

[5] <http://www-tsujii.is.s.u-tokyo.ac.jp/~genia/topics/Corpus/genia-ontology.html> We use version G3.02 of the GENIA corpus, which includes 18546 sentences (average length 9.27 sentences per article) and 490941 words (average of 26.47 words per sentence).

Figure 9.2 Dependency Structure representation in XML

```
<Psentence id="90110496-s1">
  <Pdep type="subj"  head="90110496-w4"  dep="90110496-w2"/>
  <Pdep type="aux"  head="90110496-w4"  dep="90110496-w3"/>
  <Pdep type="prep"  head="90110496-w10"  dep="90110496-w5"/>
  <Pdep type="pobj"  head="90110496-w4"  dep="90110496-w10"/>
  <Pdep type="prep"  head="90110496-w15"  dep="90110496-w11"/>
  <Pdep type="pobj"  head="90110496-w4"  dep="90110496-w15"/>
</Psentence>
```

tifier. As identifiers are preserved during all steps of processing, the existence of a unique identifier for each sentence and each token in the corpus later simplifies the task of presenting the results to the user. The second reason is that the Parmenides annotation scheme allows for a neater distinction of different 'layers' of annotations (structural, textual and conceptual) which again simplifies later steps of processing.

In DEPGENIA we have used the convention to give to each "object" of interest (e.g. sentences, tokens and domain descriptions) a unique identifier based on the concatenation of the medline identifier for the article in which the object is contained, followed by the sequential position of the object within the article (counting all objects of the same type). An example of a short GENIA article, modified in this manner, can be seen in Figure 9.4. Notice that such identifiers are not necessarily restricted to GENIA: they remain meaningful even outside that context (as long as the identification of MedLine articles via MedLine identifiers remain stable).[6]

[6]In previous experiments [Rinaldi et al., 2004c] we simply numbered the sentences sequentially according to their position in the GENIA corpus, however that approach had the disadvantage that such identifiers become meaningless outside the context of GENIA.

Table 9.1 Dependencies (in CSV format) for sentence number 90110496-s1

90110496-s1	subj	90110496-w4	associate	90110496-w2	autoantibody
90110496-s1	aux	90110496-w4	associate	90110496-w3	be
90110496-s1	prep	90110496-w10	gene	90110496-w5	with
90110496-s1	pobj	90110496-w4	associate	90110496-w10	gene
90110496-s1	prep	90110496-w15	patient	90110496-w11	in
90110496-s1	pobj	90110496-w4	associate	90110496-w15	patient

Therefore the corpus could be extended to include a larger set of MedLine abstracts, without requiring any change to the already annotated articles.[7] We then apply to the resulting modified version of GENIA a pipeline of tools defined as follows:

1. replace domain descriptions (which in GENIA are identified by a `<cons>` element) with their heads

2. lemmatization of all tokens (with morpha)[8]

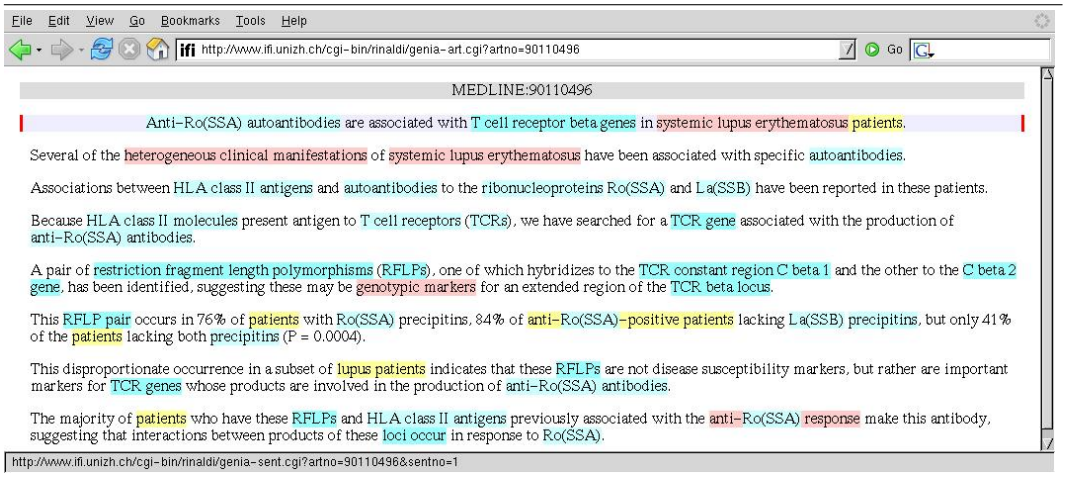
3. noun group and verb group chunking (LT CHUNK)[9]

[7] At least as long as the same tools used in GENIA are used for sentence splitting and tokenizations, or they behave exactly in the same way

[8] Available at <http://www.informatics.susx.ac.uk/research/nlp/carroll/morph.html>

[9] Available at <http://www.ltg.ed.ac.uk/software/chunk/>

Figure 9.3 Selecting a sentence



4. detection of heads in the group (with two simple rules: take the last noun from the noung group; take the last verb from the verb group)

5. dependency parsing (Pro3Gres)

The pipeline (itself declaratively specified in XML) has been implemented as an Apache Ant build file[10] which supports easy integration or replacement of specific components in the sequence. The end result of the process is a set of dependency relations, which are encoded as (sentence-id, type, head, dependent) tuples and can be delivered either in CSV or XML formats. For example Table 9.1 shows the CSV format in the case of the sentence illustrated by Figure 9.1 and Figure 9.2 the XML format. This is a format which is well suited for storage in a relational DB, for further processing with a spreadsheet tool, or for analysis with data mining algorithms.

DEPGENIA (v1.0) can be downloaded from the OntoGene web site: <http://www.ontogene.org/>. It is also possible to browse individual sen-

[10] <http://ant.apache.org/>

tences and their dependency structure in a graphical format. For convenience, we have provided a web interface that allows simplified browsing of the results, which is also accessible from the OntoGene web site (see Figure 9.3).

9.3 The Pro3Gres parser

The deep syntactic analysis builds upon the chunks using a broad-coverage probabilistic Dependency Parser [Schneider, 2003b] to identify sentence level syntactic relations between the heads of the chunks. The output is a hierarchical structure of syntactic relations - functional dependency structures. The parser (Pro3Gres [Schneider, 2003b,Schneider et al., 2004d]) uses a hand-written grammar combined with a statistical language model that calculates lexicalized attachment probabilities, similar to [Collins, 1999]. Parsing is seen as a decision process, the probability of a total parse is the product of probabilities of the individual decisions at each ambiguous point in the derivation.

Pro3Gres is a Dependency Grammar system, inspired by the works of Tesnière and Lexical-Functional Grammar (LFG) [Schneider et al., 2004c]. It is a hybrid system on many levels [Schneider et al., 2004a], combining successful parsing approaches.

Potential applications of Pro3Gres' robust, broad-coverage fast parsing of unrestricted text which are already starting to be explored include: Question Answering [Rinaldi et al., 2004a], Domain-specific customized parsing, Relation Extraction and building a domain-specific Knowledge Database [Rinaldi et al., 2004c], Parsing BioMedical Texts [Rinaldi et al., 2004c], High-Precision Parsing [Schneider et al., 2004e], or helping annotators to build Treebanks [Schneider et al., 2004e].

Two supervised models (based on Maximum Likelihood Estimations (MLE)) are used. The first is based on lexical probabilities of the heads of phrases, calculating the probability of finding specific syntactic relations (such as subject, sentential object, etc.). The second probability model is a Probabilistic Context Free Grammar (PCFG) for the production of verb phrases. Although Context Free Grammars (CFG) are alien to dependency grammar, verb phrase PCFG rules can model verb subcategorization frames which are an important component of a dependency grammar.

Probabilistic parsers generally have the advantage that they are fast and robust, and that they resolve syntactic ambiguities with high accuracy. Both of these points are prerequisites for a statistical analysis that is feasible over large amounts of text.

In comparison to shallow processing methods, parsing has the advantage that relations spanning long stretches of text can still be recognized, and that the context largely contributes to the disambiguation.

In comparison to deep linguistic, formal grammar-based parsers, however, the output of probabilistic parsers is relatively shallow, pure CFG constituency output, i.e. tree structures that do not express long distance dependencies (LDDs). In a simple example *"John wants to leave"* a shallow CFG analysis does not express the fact that John is also the implicit subject of *leave*. A parser that fails to recognize these implicit subjects, so-called control subjects, misses important information, quantitatively about 3% of all subjects.

The parser expresses distinctions that are especially important for a predicate-argument based shallow semantic representation, as far as they are expressed in the Penn Treebank training data, such as PP-attachment, most LDDs, relative clause anaphora, participles, gerunds, and the argument/adjunct distinction for NPs.

In some cases functional relations distinctions that are not expressed in the

Penn Treebank are made. Commas are e.g. disambiguated between apposition and conjunction, or the Penn tag *IN* is disambiguated between preposition and subordinating conjunction. Other distinctions that are less relevant or not clearly expressed in the Treebank are left underspecified, such as the distinction between PP arguments and adjuncts, or a number of types of subordinate clauses.

Using the Pro3Gres parser we can parse unrestricted texts including BioMedical texts at state-of-the-art performance and time cost. At about 300,000 words per hour or 2 sentences per second, its speed is between [Collins, 1999] and the [Kaplan et al., 2004] core system, while – like the latter – it can offer a deep-linguistic system. The parser is robust in that it returns the most promising set of partial structures when it fails to find a complete parse for a sentence. Since we do not analyze the (less error-prone) intra-chunk dependencies, we cannot give overall precision and recall figures, but a selective evaluation for individual dependency types, which we believe to be linguistically more informative.

A general-purpose evaluation of the parser was performed using a hand-compiled gold standard corpus [Carroll et al., 1999], which contains the grammatical relation data of 500 random sentences from a general text corpus, the Susanne corpus. The results (see [Schneider, 2003a]) can be compared to the evaluations done in [Preiss, 2003] and [Mollá and Hutchinson, 2003], and suggests that the performance of Pro3Gres is slightly above [Collins, 1999].

9.4 Evaluation

GENIA makes use of a technical, complex language: the average sentence length is 26.5 words, NP base chunk to word ratio is 2.1:1. When the termi-

nological information contained in the GENIA corpus is used (i.e. the set of domain descriptions), parsing results are competitive, even better than on general text. This indicates that, once the domain descriptions are factored out, the grammar and even the lexicalization is not very domain-specific. But, without knowledge of the descriptions, parser performance drops considerably due to mistagging and mischunking on unknown medical domain words.

The quality of the Dependency Relations has been evaluated on the basis of a set of 100 randomly selected GENIA sentences, for which the dependencies have been manually created and verified by two experts. The same 100 sentences have been extracted from DEPGENIA and automatically compared with the manual annotations. The results (for a subset of the dependency relations) is shown in the upper half of table Table 9.2.

Table 9.2 Evaluation comparing LTChunk chunking ("noterm") and near-perfect MWT knowledge ("original") on GENIA corpus

GENIA 'original'	Percentages for some relations, general, on GENIA				
	Subject	Object	noun-PP	verb-PP	subord. clause
Precision	90	94	83	82	71
Recall	86	95	82	84	75
GENIA 'noterm'	Percentages for some relations, general, on GENIA				
Precision	83	70	68	67	63
Recall	74	77	64	68	60

As a way to estimate the contribution of the terminological information (i.e.

the importance of the GENIA original annotations), we have run the parser over the test corpus, using a version containing the chunks as generated by LTCHUNK, but no information on domain descriptions ('noterm' corpus). We inspected the results of the analysis over the same 100 sentences as above. Table 9.2 show a comparison of results under those two different conditions.

More experimentally, we have integrated PP-attachment modules [Hindle and Rooth, 1993, Volk, 2002] using the GENIA corpus, because the original PP-training corpus (the Penn Treebank) is of a different domain. Against sparse data we back off to semantic GENIA classes. Our current results do not show any improvement.[11]

In order to estimate the usefulness of the dependencies for an Information Extraction task, we defined a user-level evaluation based on the usefulness of the arguments of the detected relations (their significance for a domain expert). For example, we have selected triples of the form (predicate - subject - object) and asked the expert to tell us how "good" they were. The analysis of the whole GENIA corpus resulted in 10072 such triples (records). For the evaluation of biological relevance we selected only the records containing the following predicates: *activate, bind* and *block*. This resulted in 487 records. We then removed all records where a type had not been assigned to either subject or object: this left 169 fully qualified records. This remaining set was inspected by a domain expert.

The first 'naive' evaluation was based on assigning a simple key code to each record: 'P' for positive (biologically relevant and correct, 53 cases), 'Y' for acceptable (biologically relevant but not completely correct, 102 cases) and 'N' (not biologically relevant or seriously wrong, 14 cases). This result

[11]This might be attributed to insufficient data or the relative simplicity of the GENIA Ontology.

was considered as encouraging as it showed 91.7% of relevant records.
On closer inspection of the results reported by the domain expert, we identified a number of 'typical cases', which we then asked the expert to evaluate in detail. In this second evaluation the expert had to evaluate each argument separately and mark it according to the following codes:

Y the argument is correct and informative

N the argument is completely wrong

Pr the argument is correct, but it is a pronominal reference, and it would need to be resolved to be significant (e.g. *"This protein"*)

A+ the argument is "too large" (which implies that a prepositional phrase has been erroneously attached to it)

A- the argument is "too small" (which implies that an attachment has been omitted)

We then noticed that some of the relations that had originally been considered as negative, had to be reconsidered, because our algorithm at present does not detect polarity (e.g. *"does **not** activate"*) or modality (e.g. *"**might** activate"*) and therefore some of the negative or hyphotetical cases, which the domain expert considered as incorrect, should be accepted for the purpose of the present evaluation.
Once all those points were clarified, we repeated the evaluation, which resulted in the values shown in Table 9.3. Two different types of results have been evaluated:

- how relevant are the sentences so detected wrt to the relation being investigated
- how good is the detection of the arguments of the relation

Table 9.4 shows some examples of the user-level evaluation of verb-subject-object triples extracted from the following sentences:

- *178. "Interleukin-2 (IL-2) rapidly activated Stat5 in fresh PBL, and Stat3 and Stat5 in preactivated PBL."*
- *807. "Thus, we demonstrated that IL-5 activated the Jak 2 -STAT 1 signaling pathway in eosinophils."*
- *5212. "Spi-B binds DNA sequences containing a core 5-GGAA-3 and activates transcription through this motif."*
- *16919. "The higher affinity sites bind CVZ with 20- to 50-fold greater affinity, consistent with CVZ's enhanced biological effects."*

The results reported in Table 9.3 clearly show that the biggest source of error is overexpansion of the object, plus there is a little but not insignificant problem in the detection of the subject. A close inspection of these cases points to problems with conjunctions in subject position, plus a specific problem with the construction *"does not"*.

Table 9.3 Distribution of errors

	Y	N	Pr	A+	A-
Subject	146	11	4	6	2
Object	99	1	4	59	6

The evaluation helped us to detect some systematic errors of the parser, which have been corrected in the current version. Despite those errors, the results were extremely satisfactory. Under strict correctness criteria (excluding for instance unresolved pronouns), we obtained 86.4% and 58.6% correct results in the detection of subjects and objects (respectively). Under more relaxed correctness criteria (which exclude only completely wrong arguments,

but include those partially correct) these results jumped to 93.5% and 99.4% (respectively).

More detailed results of the evaluation of the parser are reported in [Schneider et al., 2004b, Schneider et al., 2004d], while [Rinaldi et al., 2004c] describes in greater detail the applications over the GENIA corpus.

9.5 Conclusion

The types of relations between domain concepts are not limited to synonymy and hyponymy. Most of the interesting relationships are coded with syntactic means, for example as arguments in a predication. In this chapter we have introduced and evaluated the techniques that we have adopted for the task of detecting novel types of domain relations.

As a way to test the results presented in this chapter, in 2006 we participated in a collaborative text mining evaluation challenge, which specializes in biomedical literature (BioCreative), where we obtained competitive results in the task of detecting protein-protein interactions [Rinaldi et al., 2007a].

The techniques reported in this chapter have been further developed in the context of an internal research project. The results obtained have lead to a number of journal publications [Rinaldi et al., 2007b, Rinaldi et al., 2006c, Rinaldi et al., 2008].

Figure 9.4 Fragment of modified GENIA article

```
<article>
  <articleinfo>
    <bibliomisc>MEDLINE:90110496</bibliomisc>
  </articleinfo>
  <title>
    <sentence id="90110496-s1">
      <cons lex="Anti-Ro(SSA)_autoantibody"
         sem="G#protein_family_or_group">
         <w c="JJ" id="90110496-w1">Anti-Ro(SSA)</w>
         <w c="NNS" id="90110496-w2">autoantibodies</w>
      </cons>
      <w c="VBP" id="90110496-w3">are</w>
      <w c="VBN" id="90110496-w4">associated</w>
      <w c="IN" id="90110496-w5">with</w>
      <cons lex="T_cell_receptor_beta_gene"
         sem="G#DNA_family_or_group">
         <w c="NN" id="90110496-w6">T</w>
         <w c="NN" id="90110496-w7">cell</w>
         <w c="NN" id="90110496-w8">receptor</w>
         <w c="NN" id="90110496-w9">beta</w>
         <w c="NNS" id="90110496-w10">genes</w>
      </cons>
      <w c="IN" id="90110496-w11">in</w>
      <cons lex="systemic_lupus_erythematosus_patient"
         sem="G#multi_cell">
         <cons lex="systemic_lupus_erythematosus"
            sem="G#other_name">
            <w c="JJ" id="90110496-w12">systemic</w>
            <w c="NN" id="90110496-w13">lupus</w>
            <w c="NN" id="90110496-w14">erythematosus</w>
         </cons>
         <w c="NNS" id="90110496-w15">patients</w>
      </cons>
      <w c="." id="90110496-w16">.</w>
    </sentence>
  </title>
  <abstract>
    <sentence id="90110496-s2">
       [...]
    </sentence>
  </abstract>
</article>
```

Table 9.4 Some examples of extracted verb-subj-obj triples.

No	relation	subj	subj type	subj eval	obj	obj type	obj eval
178	activate	Interleukin-2 (IL-2)	#amino_acid	Y	Stat5 in fresh PBL, and Stat3 and Stat5 in preactivated PBL	#amino_acid	A+
807	activate	IL-5	#amino_acid	Y	the Jak 2 - STAT 1 signaling pathway	#other_name	Y
5212	bind	Spi-B	#amino_acid	Y	DNA sequences	#nucleic_acid	A-
16919	bind	The higher affinity sites	#other_name	Pr	CVZ with 20-	#other_organic_compound	N

Chapter 10

Conclusion

This dissertation presents a summary of research activities aimed at easing the burden of searching information within scientific and technical documentation, by exploiting the domain knowledge implicit in the technical descriptions, i.e. those textual units that act as references to specific objects of the domain.

We have tested the feasibility of using existing terminology extraction tools and algorithms for the extraction of domain descriptions over the technical manual of a modern aircraft. Although their performance is far from satisfactory on the whole, with recall far dominating precision so that the task of manually validating the results is long and tiresome, together with good visualization tools to ease the task, they have been useful in extracting a reasonably complete set of domain descriptions.

We have shown that the availability of a reliable set of domain descriptions can lead to a dramatic increase of performance in parsing technical language. We have extended an existing parser (Link Grammar) with the capability of treating all known domain descriptions as fixed syntactic units, obtaining almost 50% fewer spurious parses.

However, the advantages for the parsing process are only one of the bene-

fits that domain descriptions can bring to an NLP application for technical domains. Systems that provide access to scientific and technical documentation present the problem of "domain navigation": not all users can be expected to be completely familiar with the sublanguage of the domain. Unfamiliarity with the domain might lead to requests for information about domain objects, which contain imperfect formulations of their names (a problem which we have previously called the **Paraphrase Problem** [Rinaldi et al., 2003d]).

Even experienced users, who know the domain well, might not remember the exact form of a domain description and use a paraphrase to refer to the underlying domain concept. Besides, in the documents themselves, unless the editors have been forced to use some strict terminology control system, various paraphrases of the same compound will appear, and they need to be identified as co-referent. It becomes therefore essential to be able to detect variants of the domain descriptions used in the documents.

Despite using relatively simple techniques, we have been able to detect with high reliability not only synonyms among domain descriptions, thus giving us an instrument to master the problem of variability, but also cases of hyponymy, which can be useful in order to find descriptions which are related to the one sought for by the user, although not being strictly equivalent to it.

We have made use of the descriptions and relations so obtained within a pre-existing question answering system, improving both its performance and its usefulness to the end user: as we have shown, the usage of a structured collection of domain description in a QA system leads to an increase in the number of satisfactory answers found. In other words, the introduction of our special treatment for domain descriptions leads to a greater number of correct answers as well as to a better ranking of the answers delivered by

the system, as we have experimentally shown.

This experience leads us to conclude that a proper treatment of domain descriptions is critical for systems that deal with technical and scientific documentation. Although fully automated detection of domain description remains an unsolved bottleneck, and a degree of human intervention is needed to construct a correct and complete list of domain descriptions, other steps in the creation of a computational resource based on them, can be, to a large extent, automated. It is also the case that such a list, once created, constitutes a valuable linguistic resource which can be used for purposes other than that for which it was originally conceived.

There are numerous potential further developments of the work presented in this dissertation, some of which have been already partially pursued by the author. In particular, as we have discussed, the biomedical domain presents an interesting area of application. We have introduced techniques for making use of the domain descriptions that can be found in this domain within another extension of the question answering system previously used for our experiments, as well as within a novel type of application, focusing on relation mining.

In the new experimentation with the use of domain descriptions in the biomedical domain, the parsing techniques were further improved by the use of a parser recently developed by other members of the Zurich research team, combining linguistic and statistic techniques. In more recent work too, we have sought to extend the repertoire of relations detected between elements of the text. While the early work on the aircraft manuals was restricted to relations of synonymy and hyponymy amongst the domain descriptions, in the work in the biomedical domain we have focused on relations expressed as syntactic arguments of a simple predication.

The relations detected with this novel approach could themselves be used to

enrich a knowledge base, whose core unit are domain concepts (identified at the lexical level by the domain descriptions). In this knowledge base concepts can be correlated by various types of domain relationships, some of which can be detected automatically with the methods described in this dissertation.

All the research activities mentioned above could therefore be framed within a broader attempt at building a new generation of question answering systems, which borrows from the pioneering experiments of the 70s and 80s the idea of making use of a well structured knowledge base, but in contrast to their efforts, do not require its manual construction. The advancement of NLP techniques, and the broad availability of documents in electronic format, make it now possible to consider the automatic derivation of domain knowledge from a large enough collection of documents.

Bibliography

[Abney et al., 2000] Abney, S., Collins, M., and Singhal, A. (2000). Answer extraction. In Nirenburg, S., editor, *Proc. 6th Applied Natural Language Processing Conference*, pages 296–301, Seattle, WA. Morgan Kaufmann. 2.3.10

[Ananiadou and Tsujii, 2003] Ananiadou, S. and Tsujii, J., editors (2003). *Proceedings of the ACL 2003 Workshop on Natural Language Processing in Biomedicine*. 7.1.2.1

[ATA, 1997] ATA (1997). *ATA Common Support Data Dictionary for The Air Transport Industries*. ATA. 4.2

[Bikel et al., 1999] Bikel, D., Schwartz, R., and Weischedel, R. (1999). An algorithm that learns what's in a name. *Machine Learning - Special Issue on NL learning*, 34. 2.5

[Bobrow, 1977] Bobrow, e. a. (1977). GUS, a frame driven dialog system. In *Artificial Intelligence*, volume 8, pages 155–173. 2.1

[Breck et al., 2001] Breck, E., Burger, J., Ferro, L., Greiff, W., Light, M., Mani, I., and Rennie, J. (2001). Another sys called Qanda. In [Voorhees and Harman, 2001]. 2.3.10

[Brill et al., 2001] Brill, E., Lin, J., Banko, M., Dumais, S. T., and Ng, A. Y. (2001). Data-intensive question answering. In *Text REtrieval Conference*. 2.3.11

[Brill and Resnik, 1994] Brill, E. and Resnik, P. (1994). A rule-based approach to prepositional phrase attachment disambiguation. In *Proc. COLING '94*, volume 2, pages 998–1004, Kyoto, Japan. 2.8.2

[Bröker et al., 1994] Bröker, N., Hahn, U., and Schacht, S. (1994). Concurrent lexicalized dependency parsing: The ParseTalk model. In *Coling 94*, pages 379–385. 7.1.2.2

[Buitelaar et al., 2005] Buitelaar, P., Cimiano, P., and Magnini, B., editors (2005). *Ontology Learning and Population*. IOS Press. 10

[Burke et al., 1997] Burke, R., abd V. Kulyukin abd S. Lytinen, K. H., Tomuro, N., and Schoenberg, S. (1997). Question answering from frequently-asked question files: Experiences with the faq finder system. *AI Magazine*, 18(2):57–66. 2.2

[Carroll et al., 1999] Carroll, J., Minnen, G., and Briscoe, T. (1999). Corpus annotation for parser evaluation. In *Proceedings of the EACL-99 Post-Conference Workshop on Linguistically Interpreted Corpora*, Bergen, Norway. 9.3

[Castellví et al., 2001] Castellví, M. T. C., Bagot, R. E., and Palatresi, J. V. (2001). Automatic term detection: A review of current systems. In Bourigault, D., Jacquemin, C., and L'Homme, M.-C., editors, *Recent Advances in Computational Terminology*, pages 53–87. John Benjamins Publishing Company. 4.1

[Charniak, 1972] Charniak, E. (1972). *Towards a model of children?s story comprehension*. PhD thesis, Cambridge, Mass.: Massachusetts Institute of Technology. 2.1

[Charniak and Goldman, 1989] Charniak, E. and Goldman, R. P. (1989). A semantics for probabilistic quantifier-free first-order languages, with particular application to story understanding. In *IJCAI*, pages 1074–1079. 2.2

[Chinchor, 1998a] Chinchor, N. A. (1998a). Overview of muc-7/met-2. In *Proceedings of the Seventh Message Understanding Conference (MUC- 7)*. 1.1

[Chinchor, 1998b] Chinchor, N. A. (1998b). Overview of MUC-7/MET-2. 7.1.1

[Cimiano et al., 2005] Cimiano, P., Pivk, A., Schmidt-Thieme, L., and Staab, S. (2005). Learning taxonomic relations from heterogeneous evidence. In [Buitelaar et al., 2005]. 3.3.1

[Cimino and Barnet, 1993] Cimino, J. and Barnet, G. (1993). Automatic Knowledge Acquisition from Medline. *Methods of Information in Medicine*, 32(2):120–130. 7.1.2.1

[Cimino, 2001] Cimino, J. J. (2001). Knowledge-based terminology management in medicine. In Bourigault, D., Jacquemin, C., and L'Homme, M.-C., editors, *Recent Advances in Computational Terminology*, pages 111–126. John Benjamins Publishing Company. 7.1.1

[Clarke et al., 2001a] Clarke, C., Cormack, G., Kisman, D., and Lynam, T. (2001a). Question answering by passage selection (MultiText experiments for TREC-9). In [Voorhees and Harman, 2001]. 2.3.10

[Clarke et al., 2001b] Clarke, C., Cormack, G., Lynam, T., Li, C., and McLearn, G. (2001b). Web reinforced question answering (multitext experiments for trec 2001). In *Overview of the TREC 2001 Question Answering Track*. 2.3.11

[Collins, 1996] Collins, M. (1996). A new statistical parser based on bigram lexical dependencies. In *Proceedings of the 34st Annual Meeting of the Association for Computational Linguistics, ACL-96*, pages 184–191. 2.3.11

[Collins, 1999] Collins, M. (1999). *Head-Statistical Models for Natural Language Processing*. PhD thesis, University of Pennsylvania, Philadelphia, USA. 9.3

[Copestake et al., 1997] Copestake, A., Flickinger, D., and Sag, I. A. (1997). Minimal recursion semantics: an introduction. Technical report, CSLI, Stanford University, Stanford, CA. 2.8.3

[Craven and Kumlien, 1999] Craven, M. and Kumlien, J. (1999). Constructing biological knowledge bases by extracting information from text sources. *Proceedings of the 8th International Conference on Intelligent Systems for Molecular Biology (ISMB-99)*, -. 7.1.2.1

[Dagan et al., 2005] Dagan, I., Glickman, O., and Magnini, B. (2005). The pascal recognising textual entailment challenge. In nonero-Candela et al., Q., editor, *MLCW 2005*, volume 3944 of *LNAI*, pages 177–190. Springer-Verlag. 1.2

[Dagan et al., 1999] Dagan, I., Lee, L., and Pereira, F. (1999). Similarity-based models of cooccurrence probabilities. *Machine Learning, special issue on Natural Language Learning*, 34(1–3):43–69. 3.3.3

[Daille et al., 1996] Daille, B., Habert, B., Jacquemin, C., and Royauté, J. (1996). Empirical observation of term variations and principles for their description. *Terminology*, 3(2):197–258. 3.3.1, 5

[Dang et al., 2007] Dang, H., Lin, J., and Kelly, D. (2007). Overview of the trec 2006 question answering track. In *Proceedings of the fifteenth Text REtrieval Conference (TREC 2006)*. 2.3.8

[Daraselia et al., 2004a] Daraselia, N., Egorov, S., Yazhuk, A., Novichkova, S., Yuryev, A., and Mazo, I. (2004a). Extracting Human Protein In-

teractions from MEDLINE using a Full-Sentence Parser. *Bioinformatics*, 20(5):604–611. 7.1.2.2

[Daraselia et al., 2004b] Daraselia, N., Egorov, S., Yazhuk, A., Novichkova, S., Yuryev, A., and Mazo, I. (2004b). Extracting Protein Function Information from MEDLINE using a Full-Sentence Parser. In *Proceedings of the Second European Workshop on Data Mining and Text Mining in Bioinformatics*, Pisa, Italy. 7.1.2.2

[Dias et al., 1999] Dias, G., Guilloré, S., and Lopes, J. G. P. (1999). Multilingual aspects of multiword lexical units. In *Proceedings of the Workshop Language Technologies — Multilingual Aspects*, pages 11–21, Ljubljana. 4.2

[Dowdall et al., 2004] Dowdall, J., Lowe, W., Elleman, J., Rinaldi, F., and Hess, M. (2004). The role of multiword terminology in knowledge management. In *Proceedings of LREC 2004, Lisbon, May 24-30, 2004*, pages 915–918, Lisbon, Portugal. ELRA. (document)

[Dreyfus, 1992] Dreyfus, H. L. (1992). *What Computers Still Can't Do: A Critique of Artificial Reason*. MIT Press, Cambridge, MA, USA. 2.1

[Dumais et al., 2002] Dumais, S., Banko, M., Brill, E., Lin, J., and Ng, A. (2002). Web question answering: Is more always better. In *SIGIR' 02, Tampere, Finland*. 2.3.11

[Dyer, 1983a] Dyer, M. (1983a). *In-depth Understanding*. MIT Press. 2.1

[Dyer, 1983b] Dyer, M. (1983b). The point of thematic abstraction units. *The Behavioral and Brain Sciences*, 6(4):599–600. 2.1

[Dyer, 1983c] Dyer, M. G. (1983c). *In-Depth Understanding A Computer Model of Integrated Processing for Narrative Comprehension*. MIT Press. 2.1

[Fellbaum, 1998] Fellbaum, C. (1998). *Wordnet, an electronic lexical database*. MIT Press. 2.8.3

[Finch and Mikheev, 1997] Finch, S. and Mikheev, A. (1997). A Workbench for Finding Structure in Texts. In *Proceedings of Applied Natural Language Processing*, Washington, DC. 8.1.2

[Franzen et al., 2002] Franzen, K., Eriksson, G., Olsson, F., L.Asker, Liden, P., and Coester, J. (2002). Protein names and how to find them. *International Journal of Medical Informatics*, 67/1-3, Special issue on NLP in Biomedical Applications:49–61. 7.1.1

[Friedman et al., 2001] Friedman, C., Kra, P., Krauthammer, M., H., and Rzhetsky, A. (2001). GENIES: a Natural-Language Processing System for the Extraction of Molecular Pathways from Journal Articles. *Bioinformatics*, 17(1):74–82. 7.1.2.2

[Gaizauskas et al., 2003] Gaizauskas, R., Demetriou, G., Artymiuk, P. J., and P., W. (2003). Protein Structures and Information Extraction from Biological Texts: The PASTA System. *Bioinformatics*, 19:135–143. 7.1.2.2

[Grabar and Zweigenbaum, 2004] Grabar, N. and Zweigenbaum, P. (2004). Lexically-based terminology structuring. *Terminology, Special issue on "Recent Trends in Computational Terminology"*, 10(1):23–53. 3.3.1

[Graff, 2002] Graff, D. (2002). The aquaint corpus of english news text. 2.3.4

[Green, 1961] Green, C. e. a. (1961). BASEBALL: an automatic question answerer. In *Proceedings of the Western Joint Computer Conference*. 2.1

[Hahn et al., 2002] Hahn, U., Romacker, M., and Schulz, S. (2002). Creating Knowledge Repositories from Biomedical Reports: The medSynDiKATe

Text Mining System. In Altman, R. B., Dunker, A. K., Hunter, L., Lauderdale, K., and Klein, T. E., editors, *Proceedings of Pacific Symposium on Biocomputing*, pages 338–349, Kauai, Hawaii, USA. 7.1.2.2

[Hamon and Nazarenko, 2001] Hamon, T. and Nazarenko, A. (2001). Detection of synonymy links between terms: Experiment and results. In Bourigault, D., Jacquemin, C., and L'Homme, M.-C., editors, *Recent Advances in Computational Terminology*, pages 185–208. John Benjamins Publishing Company. 5.1

[Hanisch et al., 2003] Hanisch, D., Fluck, J., Mevissen, H.-T., and Zimmer, R. (2003). Playing biology's name game: Identifying protein names in scientific texts. In *Proc. of the Pacific Symposium on Bioinformatics (PSB)*. 7.1.1

[Harabagiu et al., 2001] Harabagiu, S., Moldovan, D., Paşca, M., Mihalcea, R., Surdeanu, M., Bunescu, R., Gîrju, R., Rus, V., and Morarescu, P. (2001). FALCON: Boosting knowledge for answer engines. In [Voorhees and Harman, 2001]. 2.3.11

[Harabagiu et al., 2000] Harabagiu, S., Moldovan, D., Pasca, M., Mihalcea, R., Surdeanu, M., Bunescu, R., and Girju, R. (2000). FALCON: Boosting Knowledge for Answer Engines. In *Proceedings of the Text Retrieval Conference*, Gaithersburg, MD. 2.3.2, 2.4

[Harman and Candela, 1989] Harman, D. K. and Candela, G. T. (1989). A very fast prototype retrieval using statistical ranking. *SIGIR Forum*, 23(3/4):100–110. 2.8.4

[Harris, 1968] Harris, Z. (1968). *Mathematical Structures of Language*. Wiley, New York. 3.3.3

[Hearst, 1994] Hearst, M. (1994). Multi-paragraph segmentation of expository text. In *32nd. Annual Meeting of the Association for Computational Linguistics*, pages 9–16, New Mexico State University, Las Cruces, New Mexico. 2.5

[Hearst, 1992] Hearst, M. A. (1992). Automatic acquisition of hyponyms from large text corpora. In *Proceedings of COLING '92, Nantes*, pages 539–545. 3.3.2

[Hermjakob, 2001] Hermjakob, U. (2001). Parsing and question classification for question answering. In *Proc. of the ACL'01 workshop "Open-Domain Question Answering"*, pages 17–22. 2.3.10

[Hermjakob, 2002] Hermjakob, U. (2002). Qtargets used in webclopedia. Last retrieved 15th Nov 2006. 2.5

[Hermjakob and Mooney, 1997] Hermjakob, U. and Mooney, R. (1997). Learning parse and translation decisions from examples with rich context. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics*, pages 482– 489. 2.5

[Hess et al., 2002] Hess, M., Dowdall, J., and Rinaldi, F. (2002). The challenge of technical text. In *LREC-2002: Workshop on Question Answering: Strategy and Resources*, pages 27–34, Las Palmas. 1

[Hindle, 1990] Hindle, D. (1990). Noun-classification from predicate-argument structures. In *Proceedings of the 28th Annual Meeting of the Association of Computational Linguistics (ACL-1990)*, pages 268–275. 3.3.3

[Hindle and Rooth, 1993] Hindle, D. and Rooth, M. (1993). Structural ambiguity and lexical relations. *Computational Linguistics*, 19:103–120. 9.4

[Hirschman and Gaizauskas, 2001a] Hirschman and Gaizauskas (2001a). Natural Language QA, the view from here. *Natural Language Engineering*, 7(4). 1.3

[Hirschman and Gaizauskas, 2001b] Hirschman, L. and Gaizauskas, R. (2001b). Natural language question answering: The view from here. *Natural Language Engineering*, 7(5):275–300. 2.3.10

[Hobbs, 1985] Hobbs, J. R. (1985). Ontological promiscuity. In *Proc. ACL'85*, pages 61–69. University of Chicago, Association for Computational Linguistics. 2.8.3

[Hobbs et al., 1993] Hobbs, J. R., Stickel, M. E., Appelt, D. E., and Martin, P. (1993). Interpretation as abduction. *Artif. Intell.*, 63(1-2):69–142. 2.2

[Hobbs et al., 1988] Hobbs, J. R., Stickel, M. E., Martin, P., and Edwards, D. (1988). Interpretation as abduction. In *ACL*, pages 95–103. 2.2

[Hovy et al., 2000] Hovy, E., Gerber, L., Hermjakob, U., Junk, M., and Lin, C.-Y. (2000). Question answering in webclopedia. In *Ninth Text REtrieval Conference*, Gaithersburg, MD. 2.1, 2.5

[Hovy et al., 2001] Hovy, E., Gerber, L., Hermjakob, U., Lin, C.-Y., and Ravichandran, D. (2001). Toward semantics-based answer pinpointing. In *Proceedings of the Human Language Technology Conference, (HLT2001), San Diego, CA, U.S.A., March 18-21, 2001*. 2.5

[Hovy et al., 2002] Hovy, E., Hermjakob, U., and Ravichandran, D. (2002). A question/answer typology with surface text patterns. In *Proceedings of the DARPA Human Language Technology Conference (HLT). San Diego, CA.* 2.5

[Humphreys et al., 2001] Humphreys, K., Gaizauskas, R., Hepple, M., and Sanderson, M. (2001). University of Sheffield TREC-8 Q&A System. In *[Voorhees and Harman, 2000]*. 2.3.1

[Ibekwe-Sanjuan, 1998] Ibekwe-Sanjuan, F. (1998). Terminological Variation, a Means of Identifying Research Topics from Texts. In *Proceedings of COLING-ACL*, pages 571–577, Quebec,Canada. 3.3.1, 5

[Ibekwe-SanJuan and Dubois, 2002] Ibekwe-SanJuan, F. and Dubois, C. (2002). Can Syntactic Variations Highlight Semantic Links Between Domain Topics? In *Proceedings of the 6th International Conference on Terminology and Knowledge Engineering (TKE02)*, pages 57–64, Nancy. 5

[iso1087, 2000] iso1087 (2000). *ISO 1087-1:2000, Terminology work – Vocabulary – Part 1: Theory and application*. International Organization for Standardization. 3.2

[Jacquemin, 2001] Jacquemin, C. (2001). *Spotting and Discovering Terms through Natural Language Processing*. MIT Press. 3.3.1, 5, 5.1

[Johnson, 2002a] Johnson, M. (2002a). A simple pattern-matching algorithm for recovering empty nodes and their antecedents. In *Proceedings of the 40th Meeting of the ACL*, University of Pennsylvania, Philadelphia. 8.1.3

[Johnson, 2002b] Johnson, S., editor (2002b). *Proceedings of the ACL 2002 Workshop on Natural Language Processing in the Biomedical domain*. 7.1.2.1

[Kageura, 2002] Kageura, K. (2002). *The Dynamics of Terminology, A descriptive theory of term formation and terminological growth*. Terminology and Lexicography, Research and Practice. John Benjamins Publishing. 2

[Kaplan et al., 2004] Kaplan, R., Riezler, S., King, T. H., III, J. T. M., Vasserman, A., and Crouch, R. (2004). Speed and accuracy in shallow and deep stochastic parsing. In *Proceedings of HLT/NAACL 2004*, Boston, MA. 9.3

[Katz, 1997] Katz, B. (1997). From sentence processing to information access on the world wide web. In *AAAI Spring Symposium on Natural Language Processing for the World Wide Web*, Stanford, California. 2.6

[Katz et al., 2005a] Katz, B., Borchardt, G., and Felshin, S. (2005a). Syntactic and semantic decomposition strategies for question answering from multiple resources. In *Proceedings of the AAAI 2005 Workshop on Inference for Textual Question Answering, Pittsburgh, PA.*, pages 35–41. 2.6

[Katz et al., 2003] Katz, B., Lin, J., Loreto, D., Hildebrandt, W., Bilotti, M., Felshin, S., Fernandes, A., Marton, G., , and Mora, F. (2003). Integrating web-based and corpus-based techniques for question answering. In *Proceedings of the Twelfth Text REtrieval Conference (TREC 2003), Gaithersburg, Maryland*. 2.6

[Katz et al., 2005b] Katz, B., Marton, G., Borchardt, G., Brownell, A., Felshin, S., Loreto, D., Louis-Rosenberg, J., Lu, B., Mora, F., Stiller, S., Uzuner, O., and Wilcox, A. (2005b). External knowledge sources for question answering. In *Proceedings of the 14th Annual Text REtrieval Conference (TREC2005), Gaithersburg, MD.* 2.6

[KDD, 2002] KDD (2002). Kdd cup competition web site. `http://www.biostat.wisc.edu/~craven/kddcup/`. 7.1.3

[Kim et al., 2001] Kim, H., Kim, K., Lee, G. G., and Seo, J. (2001). Maya: A fast question-answering system based on a predicate answer indexer. In *Proc. of the ACL'01 workshop "Open-Domain Question Answering"*, pages 9–16. 2.3.10

[Kim et al., 2003a] Kim, J., Ohta, T., Tateisi, Y., and Tsujii, J. (2003a). GENIA corpus - a semantically annotated corpus for bio-textmining. *Bioinformatics*, 19(1):i180–i182.

[Kim et al., 2003b] Kim, J., Ohta, T., Tateisi, Y., and Tsujii, J. (2003b). GENIA Corpus - a Semantically Annotated Corpus for Bio-Textmining. *Bioinformatics*, 19(1):180–182. 8.1, 9.2

[Koike and Takagi, 2004] Koike, A. and Takagi, T. (2004). Gene/protein/family name recognition in biomedical literature. In *Proc. of NAACL/HLT 2004 BioLINK Workshop*, pages 9–16, Boston, MA. 7.1.1

[Kolodner, 1980] Kolodner, J. L. (1980). Organizing memory and keeping it organized. In *AAAI*, pages 331–333. 2.1

[Lappin and Leass, 1994] Lappin, S. and Leass, H. J. (1994). An algorithm for pronominal anaphora resolution. *Computational Linguistics*, 20(4):535–561. 2.8.2

[Lehnert, 1978] Lehnert, W. (1978). *The Process of Question Answering*. New Jersey. 2.1

[Lehnert, 1982] Lehnert, W. G. (1982). Plot units: A narrative summarization strategy. In Lehnert, W. G. and Ringle, M. H., editors, *Strategies for natural language processing*. Hillsdale, NJ: Lawrence Erlbaum Associates. 2.1

[Lenat, 1995] Lenat, D. B. (1995). Cyc: A large-scale investment in knowledge infrastructure. *Communications of the ACM*, 11:33–38. 10

[Lin, 1998] Lin, D. (1998). Automatic retrieval and clustering of similar words. In *Proceedings of the 36th Annual Meeting of the Association of Com-*

putational Linguistics and the 17th International Conference on Computational Linguistics (COLING-ACL '98), pages 768–774, Montreal. 3.3.3, 3.3.3

[Lin, 2002] Lin, J. (2002). The web as a resource for question answering: Perspectives and challenges. In *Proceedings of the Third International Conference on Language Resources and Evaluation*, pages 2120–2127, Las Palmas, Spain. 2.3.11, 2.6

[Linden and Piitulainen, 2004] Linden, K. and Piitulainen, J. (2004). Discovering synonyms and other related words. In *Proceedings of the 3rd International Workshop on Computational Terminology (CompuTerm 2004)*, Geneva, Switzerland. 3.3.3

[Marie-Reinberger and Spyns, 2005] Marie-Reinberger, L. and Spyns, P. (2005). Discovering knowledge in texts for the learning of dogma-inspired ontologies. In [Buitelaar et al., 2005]. 3.3.1

[Martin, 1990] Martin, J. H. (1990). *A Computational Model of Metaphor Interpretation*. Academic Press. ISBN 0-12-474730-2. 2.2

[McCord et al., 1992] McCord, M., Bernth, A., Lappin, S., and Zadrozny, W. (1992). Natural language processing within a slot grammar framework. *International Journal on Artificial Intelligence Tools*, 1(2):229–277. 2.8.2

[Meyers et al., 1998] Meyers, A., Macleod, C., Yangarber, R., Grishman, R., Barrett, L., and Reeves, R. (1998). Using NOMLEX to produce nominalization patterns for information extraction. In *Proceedings: the Computational Treatment of Nominals, Montreal, Canada, (Coling-ACL98 workshop)*. 2.8.2

[Minsky, 1975] Minsky, M. (1975). A framework for representing knowledge. In Winston, P. J., editor, *The psychology of computer vision*, pages 211–277. McGraw-Hill, New York, NY. 2.1

[Miyao et al., 2004] Miyao, Y., Ninomiya, T., and Tsujii, J. (2004). Corpus-oriented grammar development for acquiring a Head-driven Phrase Structure Grammar from the Penn Treebank. In *Proceedings of IJCNLP-04*. 7.1.2.2

[Moldovan et al., 2002] Moldovan, D., Harabagiu, S., Girju, R., Morarescu, P., Lacatusu, F., Novischi, A., Badulescu, A., and Bolohan, O. (2002). LCC Tools for Question Answering. In *Proceedings of TREC-11*. 2.3.4

[Moldovan et al., 2000] Moldovan, D., Harabagiu, S., Pasca, M., and Mihalcea, R. (2000). Lasso: A tool for surfing the answer net. In [Voorhees, 1999], pages 175–184. 2.3.1, 2.4, 2.5

[Molla, 2000] Molla, D. (2000). *ExtrAns: An Answer Extraction System for Unix Manpages*. IFI, CL, University of Zurich. 2.8, 2.2

[Mollá, 2001] Mollá, D. (2001). Ontologically promiscuous flat logical forms for NLP. In Bunt, H., van der Sluis, I., and Thijsse, E., editors, *Proceedings of IWCS-4*, pages 249–265. Tilburg University. 2.8.3

[Mollá and Hutchinson, 2003] Mollá, D. and Hutchinson, B. (2003). Intrinsic versus extrinsic evaluations of parsing systems. In *Proceedings of EACL03 workshop on Evaluation Initiatives in Natural Language Processing*, pages 43–50, Budapest. 7.1.2.2, 9.3

[Mollá et al., 2003a] Mollá, D., Rinaldi, F., Schwitter, R., Dowdall, J., and Hess, M. (2003a). Answer Extraction from Technical Texts. *IEEE Intelligent Systems*, 18(4):12–17. (document), 2.8.1

[Mollá et al., 2000a] Mollá, D., Schneider, G., Schwitter, R., and Hess, M. (2000a). Answer Extraction using a Dependency Grammar in ExtrAns. *Traitement Automatique de Langues (T.A.L.), Special Issue on Dependency Grammar*, 41(1):127–156. . 2.8.1, 2.8.1, 2.8.2, 2.8.3, 2.8.3

[Mollá et al., 2000b] Mollá, D., Schwitter, R., Hess, M., and Fournier, R. (2000b). ExtrAns, an answer extraction system. *T.A.L. special issue on Information Retrieval oriented Natural Language Processing*, pages 495–522. . 2.8.1, 2.8.2

[Mollá et al., 2003b] Mollá, D., Schwitter, R., Rinaldi, F., Dowdall, J., and Hess, M. (2003b). NLP for Answer Extraction in Technical Domains. In *Proc. of EACL 03 Workshop: Natural Language Processing for Question Answering*, pages 5–11, Budapest. (document)

[Morin and Jacquemin, 2004] Morin, E. and Jacquemin, C. (2004). Automatic acquisition and expansion of hypernym links. *Computer and the Humanities*, 38(4):343–362. 3.3.2

[Navigli et al., 2005] Navigli, R., Velardi, P., Cucchiarelli, A., and Neri, F. (2005). Quantitative and qualitative evaluation of the ontolearn ontology learning system. In [Buitelaar et al., 2005]. 3.3.1

[Nédellec and Nazarenko, 2005] Nédellec, C. and Nazarenko, A. (2005). Ontology-based information extraction vs ontology refinement based on information extraction. In [Buitelaar et al., 2005]. 3.3.1

[Nenadic et al., 2004] Nenadic, G., Spasic, I., and Ananiadou, S. (2004). Mining term similarities from corpora. *Terminology, Special issue on "Recent Trends in Computational Terminology"*, 10(1):55–81. 3.3.1

[Niu et al., 2003] Niu, Y., Hirst, G., McArthur, G., and Rodriguez-Gianolli, P. (2003). Answering clinical questions with role identification. In Ananiadou, S. and Tsujii, J., editors, *Proceedings of the ACL 2003 Workshop on Natural Language Processing in Biomedicine*, pages 73–80. 7

[Norvig, 1983] Norvig, P. (1983). Frame activated inferences in a story understanding program. In *IJCAI*, pages 624–626. 2.2

[Norvig, 1987] Norvig, P. (1987). *A Unified Theory of Inference for Text Understanding*. PhD thesis, University of California, Berkeley, Computer Science Department. 2.2

[Novichkova et al., 2003] Novichkova, S., Egorov, S., and Daraselia, N. (2003). MedScan, a Natural Language Processing Engine for MEDLINE Abstracts. *Bioinformatics*, 19(13):1699–1706. 7.1.2.2

[Okazaki and Ananiadou, 2006] Okazaki, N. and Ananiadou, S. (2006). Building an abbreviation dictionary using a term recognition approach. *Bioinformatics*, 22(24):3089–3095. 2

[Ono et al., 2001] Ono, T., Hishigaki, H., Tanigami, A., and Takagi, T. (2001). Automated Extraction of Information on Protein-Protein Interactions from the Biological Literature. *Bioinformatics*, 17(2):155–161. 7.1.2.1

[Park et al., 2001] Park, J. C., Kim, H. S., and Kim, J. J. (2001). Bidirectional Incremental Parsing for Automatic Pathway Identification with Combinatory Categorial Grammar. In *Proceedings of Pacific Symposium on Biocomputing (PSB)*, Big Island, Hawaii, USA. 7.1.2.2

[Park et al., 2003] Park, Y., Byrd, R. J., and Boguraev, B. K. (2003). Towards Ontologies on Demand. In *Proceedings of the Workshop on Semantic Web*

Technologies for Searching and Retrieving Scientific Data (ISWC-03), Florida, USA. 7.1.1

[Pasca and Harabagiu, 2001] Pasca, M. and Harabagiu, S. (2001). Answer mining from on-line documents. In *Proc. of the ACL'01 workshop "Open-Domain Question Answering"*, pages 38–45. 2.3.10

[Preiss, 2003] Preiss, J. (2003). Using grammatical relations to compare parsers. In *Proc. of EACL 03*, Budapest, Hungary. 9.3

[Pum-Ryu and Key-Choi, 2005] Pum-Ryu, M. and Key-Choi, S. (2005). Measuring the specificity of terms for automatic hierarchy construction. In [Buitelaar et al., 2005]. 3.3.1

[Pustejovsky et al., 2002] Pustejovsky, J., Castaño, J., Zhang, J., Cochran, B., and Kotecki, M. (2002). Robust Relational Parsing over Biomedical Literature: Extracting Inhibit Relations. In *Pacific Symposium on Biocomputing*, pages 362–385. 7.1.2.2

[Quillian, 1968] Quillian, M. R. (1968). Semantic memory. In Minsky, M., editor, *Semantic Information Processing*. Massachusetts Institute of Technology Press, Cambridge. 2.2

[Ravichandran and Hovy, 2002] Ravichandran, D. and Hovy, E. (2002). Learning surface text patterns for a question answering system. In *Proceedings of the 40th ACL conference. Philadelphia, PA*. 2.5

[Rieger, 1975] Rieger, C. J. (1975). Conceptual overlays: A mechanism for the interpretation of sentence meaning in context. In *IJCAI*, pages 143–150. 2.1

[Rinaldi et al., 2003a] Rinaldi, F., Dowdall, J., Hess, M., Ellman, J., Zarri, G. P., Persidis, A., Bernard, L., and Karanikas, H. (2003a). Multilayer

Annotations in PARMENIDES. In Handschuh, S., Koivunen, M., Dieng, R., and Staab, S., editors, *The K-CAP2003 workshop on "Knowledge Markup and Semantic Annotation", Sanibel, Florida*, pages 33–40. 9.2

[Rinaldi et al., 2003b] Rinaldi, F., Dowdall, J., Hess, M., Kaljurand, K., and Karlsson, M. (2003b). The Role of Technical Terminology in Question Answering. In *Proceedings of TIA-2003, Terminologie et Intelligence Artificielle*, pages 156–165, Strasbourg. (document), 4.1, 5.3, 6.1

[Rinaldi et al., 2002a] Rinaldi, F., Dowdall, J., Hess, M., Kaljurand, K., Koit, M., Vider, K., and Kahusk, N. (2002a). Terminology as Knowledge in Answer Extraction. In *Proceedings of the 6th International Conference on Terminology and Knowledge Engineering (TKE02)*, pages 107–113, Nancy. (document), 1.4, 4.4, 5.1, 5.3, 6.1

[Rinaldi et al., 2002b] Rinaldi, F., Dowdall, J., Hess, M., Mollá, D., and Schwitter, R. (2002b). Towards Answer Extraction: an application to Technical Domains. In *ECAI2002, European Conference on Artificial Intelligence, Lyon*, pages 460–464. (document)

[Rinaldi et al., 2003c] Rinaldi, F., Dowdall, J., Hess, M., Mollá, D., Schwitter, R., and Kaljurand, K. (2003c). Knowledge-Based Question Answering. In Palade, V., Howlett, R. J., and Jain, L., editors, *Proceedings of KES-2003, Knowledge-Based Intelligent Information and Engineering Systems*, volume 2773 of *Lecture Notes in Artificial Intelligence*, pages 780–792, Oxford. Springer-Verlag. (document), 5.3

[Rinaldi et al., 2003d] Rinaldi, F., Dowdall, J., Kaljurand, K., Hess, M., and Mollá, D. (2003d). Exploiting paraphrases in a question answering system. In *The ACL-2003 workshop on Paraphrasing (IWP2003), July 2003, Sapporo, Japan*. (document), 10

[Rinaldi et al., 2004a] Rinaldi, F., Dowdall, J., Schneider, G., and Persidis, A. (2004a). Answering Questions in the Genomics Domain. In *The ACL 2004 workshop on Question Answering in Restricted Domains, Barcelona, July 2004*. (document), 9.3

[Rinaldi et al., 2004b] Rinaldi, F., Hess, M., Dowdall, J., Mollá, D., and Schwitter, R. (2004b). Question answering in terminology-rich technical domains. In Maybury, M. T., editor, *New Directions in Question Answering*, pages 71–82. MIT/AAAI Press. (document)

[Rinaldi et al., 2002c] Rinaldi, F., Hess, M., Mollá, D., Schwitter, R., Dowdall, J., Schneider, G., and Fournier, R. (2002c). Answer extraction in technical domains. In Gelbukh, A., editor, *Computational Linguistics and Intelligent Text Processing*, volume 2276 of *Lecture Notes in Computer Science*, pages 360–369. Springer-Verlag. (document), 1, 2.5

[Rinaldi et al., 2008] Rinaldi, F., Kappeler, T., Kaljurand, K., Schneider, G., Klenner, M., Clematide, S., Hess, M., von Allmen, J.-M., Parisot, P., Romacker, M., and Vachon, T. (2008). OntoGene in BioCreative II. *Genome Biology*, 9(Suppl 2):S13. 9.5

[Rinaldi et al., 2007a] Rinaldi, F., Kappeler, T., Kaljurand, K., Schneider, G., Klenner, M., Hess, M., von Allmen, J.-M., Romacker, M., and Vachon, T. (2007a). OntoGene in Biocreative II. In *Proceedings of the II Biocreative Workshop*. 9.5

[Rinaldi et al., 2004c] Rinaldi, F., Schneider, G., Kaljurand, K., Dowdall, J., Andronis, C., Persidis, A., and Konstandi, O. (2004c). Mining relations in the GENIA corpus. In Scheffer, T., editor, *Second European Workshop on Data Mining and Text Mining for Bioinformatics*, pages 61–68. ECML/PKDD. 6, 9.3, 9.4

[Rinaldi et al., 2006a] Rinaldi, F., Schneider, G., Kaljurand, K., and Hess, M. (2006a). Tools for text mining over biomedical literature. In *Proceedings of the ECAI Conference, poster session*. 9.1

[Rinaldi et al., 2007b] Rinaldi, F., Schneider, G., Kaljurand, K., Hess, M., Andronis, C., Konstandi, O., and Persidis, A. (2007b). Mining of Functional Relations between Genes and Proteins over Biomedical Scientific Literature using a Deep-Linguistic Approach. *Journal of Artificial Intelligence in Medicine*, 39:127–136. 9.1, 9.5

[Rinaldi et al., 2006b] Rinaldi, F., Schneider, G., Kaljurand, K., Hess, M., and Romacker, M. (2006b). An Environment for Relation Mining over Richly Annotated Corpora: the case of GENIA. In *Proceedings of SMBM 2006, Second International Symposium on Semantic Mining in Biomedicine*, pages 68–75, Jena, Germany. 9.1

[Rinaldi et al., 2006c] Rinaldi, F., Schneider, G., Kaljurand, K., Hess, M., and Romacker, M. (2006c). An Environment for Relation Mining over Richly Annotated Corpora: the case of GENIA. *BMC Bioinformatics*, 7(Suppl 3):S3. 9.1, 9.5

[Rinaldi and Yuste, 2003] Rinaldi, F. and Yuste, E. (2003). Extracción automática de respuestas para documentacion técnica. In *SEPLN 2003 (XIX Congreso de la Sociedad Española para el Procesamiento del Lenguaje Natural), Alcalá de Henares (Madrid), Spain*. (document)

[Rinaldi et al., 2005] Rinaldi, F., Yuste, E., Schneider, G., Hess, M., and Roussel, D. (2005). Exploiting technical terminology for knowledge management. In [Buitelaar et al., 2005]. (document)

[Rindflesch et al., 2000] Rindflesch, T., Tanabe, L., Weinstein, J. N., and Hunter, L. (2000). Edgar: Extraction of drugs, genes and relations from

the biomedical literature. In *Pacific Symposium on Biocomputing*, pages 514–25. 7.1.2.1

[Sackett et al., 2000] Sackett, D. L., Straus, S. E., Richardson, W. S., Rosenberg, W., and Haynes, R. B. (2000). *Evidence Based Medicine: How to Practice and Teach EBM*. Churchill Livingstone. 7

[Sampo et al., 2004] Sampo, P., Ginter, F., Pahikkala, T., Boberg, J., Järvinen, J., Salakoski, T., and Koivula, J. (2004). Analysis of Link Grammar on Biomedical Dependency Corpus Targeted at Protein-Protein Interactions. In *Proceedings of Coling 04 Workshop on Natural Language Proc essing in Biomedicine and its Applications (NLPBA/BioNLP)*, Geneva, Switzerland. 7.1.2.2

[Schank, 1975] Schank, R. (1975). *Conceptual Information Processing*. The Netherlands. 2.1

[Schank and Abelson, 1977] Schank, R. and Abelson, R. (1977). *Scripts, Plans, goals, and understanding: An enquiry into human knowledge structures*. Lawrence Erlbaum Associates. 2.1

[Schank, 1972] Schank, R. C. (1972). Conceptual dependency: A theory of natural language understanding. *Cognitive Psychology*, 3(4):pages 532–631. 2.1

[Schneider, 2003a] Schneider, G. (2003a). A low-complexity, broad-coverage probabilistic Dependency Parser for English. In *Proceedings of NAACL/HLT Student session*, Edmonton, Canada. 9.3

[Schneider, 2003b] Schneider, G. (2003b). Extracting and Using Trace-Free Functional Dependencies from the Penn Treebank to Reduce Parsing

Complexity. In Nivre, J. and Hinrichs, E., editors, *Proceedings of The Second Workshop on Treebanks and Linguistic Theories (TLT 2003)*, volume 9 of *Mathematical Modelling in Physics, Engineering and Cognitive Sciences*, pages 153–164, Växjö, Sweden. 8.1.3, 9.1, 9.3

[Schneider et al., 1999] Schneider, G., Aliod, D. M., and Hess, M. (1999). Inkrementelle minimale logische formen für die antwortextraktion. Proceedings of 4th Linguistic Colloquium, University of Mainz, FASK, September 7-10, 1999. 2.8.3

[Schneider et al., 2004a] Schneider, G., Dowdall, J., and Rinaldi, F. (2004a). A robust and deep-linguistic theory applied to large-scale parsing. In *Coling 2004 Workshop on Robust Methods in the Analysis of Natural Language Data (ROMAND 2004)*, Geneva, Switzerland, August 2004. 9.3

[Schneider et al., 2004b] Schneider, G., Dowdall, J., and Rinaldi, F. (2004b). A robust and hybrid deep-linguistic theory applied to large scale parsing. In Pallotta, V. and Todirascu, A., editors, *COLING-2004 workshop on Robust Methods in Analysis of Natural language Data*, pages 14–23, Geneva, Switzerland. 9.4

[Schneider et al., 2004c] Schneider, G., Rinaldi, F., and Dowdall, J. (2004c). Fast, deep-linguistic statistical dependency parsing. In *Coling 2004 Workshop on Recent Advances in Dependency Grammar*, Geneva, Switzerland, August 2004. 8.1.3, 9.3

[Schneider et al., 2004d] Schneider, G., Rinaldi, F., and Dowdall, J. (2004d). Fast, Deep-Linguistic Statistical Minimalist Dependency Parsing. In Kruijff, G. and Duchier, D., editors, *COLING-2004 workshop on Recent Advances in Dependency Grammars, August 2004, Geneva, Switzerland*, pages 33–40. 9.3, 9.4

[Schneider et al., 2004e] Schneider, G., Rinaldi, F., Kaljurand, K., and Hess, M. (2004e). Steps towards a GENIA Dependency Treebank. In *Proceedings of Treebanks and Linguistic Theories (TLT) 2004*, Tübingen, Germany, December 2004. accepted. 9.3

[Schütze, 1998] Schütze, H. (1998). Automatic word sense discrimination. *Computational Linguistics*, 24(1):97–124. 3.3.3

[Sekimizu et al., 1998] Sekimizu, T., Park, H., and Tsujii, J. (1998). Identifying the interaction between genes and gene products based on frequently seen verbs in Medline abstracts. In *Proccedings of the ninth Workshop on Genome Informatics (GIW 98). Universal Academy Press*, pages 62–71. 7.1.2.1

[Shatkay and Feldman, 2003] Shatkay, H. and Feldman, R. (2003). Mining the biomedical literature in the genomic era: An overview. *Journal of Computational Biology*, 10 (3):821–855. 7.1.2.2

[Simmons, 1965] Simmons, R. (1965). Answering english questions by computer: a survey. *Communications of the ACM*, 8(1):53–70. 2.1

[Simmons, 1973] Simmons, R. (1973). *Semantic Networks: computation and use for understanding English sentences*, chapter Computer Models of Thought and Language. San Francisco. 2.1

[Sleator and Temperley, 1991] Sleator, D. and Temperley, D. (1991). Parsing English with a Link Grammar. Technical Report Technical Report CMU-CS-91-196, Carnegie Mellon University Computer Science. 7.1.2.2

[Sleator and Temperley, 1993] Sleator, D. and Temperley, D. (1993). Parsing english with a link grammar. In *Third International Workshop on parsing Technologies*. 2.8.2, 2.8.3, 6.1

[Spasić et al., 2003] Spasić, I., Nenadić, G., and Ananiadou, S. (2003). Using domain-specific verbs for term classification. In Ananiadou, S. and Tsujii, J., editors, *Proceedings of the ACL 2003 Workshop on Natural Language Processing in Biomedicine*, pages 17–24. 7.1.1

[Srihari and Li, 2000] Srihari, R. and Li, W. (2000). Information extraction supported question answering. In [Voorhees, 1999], pages 185–196. 2.3.1, 2.5

[Stapley and Benoit, 2000] Stapley, B. and Benoit, G. (2000). Bibliometrics: information retrieval and visualization from co-occurrence of gene names in MedLine abstracts. In *Proceedings of the Pacific Symposium on Biocomputing (Oahu, Hawaii)*, pages 529–540. 7, 7.1.1

[Subbotin and Subbotin, 2001] Subbotin, M. and Subbotin, S. (2001). Patterns of potential answer expressions as clues to the right answers. In *Proceedings of TREC 2001*. 2.3.3

[Sutcliffe et al., 1996] Sutcliffe, R. F. E., Koch, H.-D., and McElligott, A., editors (1996). Rodopi, Amsterdam. 10

[Sutcliffe and McElligott, 1996a] Sutcliffe, R. F. E. and McElligott, A. (1996a). Using the link parser of Sleator and Temperley to analyse a software manual corpus. In [Sutcliffe et al., 1996], chapter 6, pages 89–102. 1.1

[Sutcliffe and McElligott, 1996b] Sutcliffe, R. F. E. and McElligott, A. (1996b). Using the link parser of Sleator and Temperley to analyse a software manual corpus. In [Sutcliffe et al., 1996], chapter 6, pages 89–102. 6.1

[TREC-GE, 2003] TREC-GE (2003). Trec genomics track - roadmap. http://medir.ohsu.edu/~genomics/roadmap.html. 7, 7.1.3

[Volk, 2002] Volk, M. (2002). Combining unsupervised and supervised methods for PP-attachment disambiguation. In *Proceedings of COLING 2002, Taipeh*. 9.4

[Voorhees, 1999] Voorhees, E. M. (1999). The TREC-8 Question Answering Track Report. In *Proceedings of TREC-8*, pages 77–83. 10

[Voorhees, 2000a] Voorhees, E. M. (2000a). The TREC-8 Question Answering Track Evaluation. In [Voorhees and Harman, 2000]. 1.1

[Voorhees, 2000b] Voorhees, E. M. (2000b). The TREC-8 Question Answering Track Report. In [Voorhees and Harman, 2000]. 2.3.11

[Voorhees, 2001] Voorhees, E. M. (2001). The TREC question answering track. *Natural Language Engineering*, 7(4):361–378. 2.3, 2.3.2, 2.3.10

[Voorhees, 2003] Voorhees, E. M. (2003). Overview of the TREC 2003 question answering track. In *Text REtrieval Conference*. 2.3.4, 2.3.5

[Voorhees, 2004] Voorhees, E. M. (2004). Overview of the TREC 2004 question answering track. In *Text REtrieval Conference*. 2.3.6

[Voorhees and Harman, 2000] Voorhees, E. M. and Harman, D., editors (2000). *The Eighth Text REtrieval Conference (TREC-8)*. NIST. 10

[Voorhees and Harman, 2001] Voorhees, E. M. and Harman, D., editors (2001). *Proceedings of the Ninth Text REtrieval Conference (TREC-9), Gaithersburg, Maryland, November 13-16, 2000*. 1.1, 2.3.11, 10

[Voorhees and Tice, 1999] Voorhees, E. M. and Tice, D. M. (1999). The TREC-8 Question Answering Track evaluation. In *Proceedings of the 8th Text Retrieval Conference, NIST, 2000*, pages 77–83. 2.3.11

[Weeds, 2003] Weeds, J. (2003). *Measures and Applications of Lexical Distributional Similarity*. Ph.D. thesis, Department of Informatics, University of Sussex. 3.3.3

[Weeds et al., 2005] Weeds, J., Dowdall, J., Schneider, G., Keller, B., and Weir, D. (2005). Using distributional similarity to organise BioMedical terminology. *Terminology*, 11(1). To appear. 3.3.3

[Weizenbaum, 1966] Weizenbaum, J. (1966). Eliza - a computer program for the study of natural language communication between man and machine. *Communications of the Association for Computing Machinery*, 9:36–45. 2.1

[Wilensky, 1983] Wilensky, R. (1983). Memory and inference. In *IJCAI*, pages 402–404. 2.2

[Wilensky, 1987] Wilensky, R. (1987). Ther berkley unix consultant project. In *Wissensbasierte Systeme*, pages 286–296. 2.2

[Wilensky et al., 1984] Wilensky, R., Arens, Y., and Chin, D. (1984). Talking to unix in english: an overview of uc. *Commun. ACM*, 27(6):574–593. 2.2

[Wilensky et al., 1988] Wilensky, R., Chin, D. N., Luria, M., Martin, J. H., Mayfield, J., and Wu, D. (1988). The berkeley UNIX consultant project. *Computational Linguistics*, 14(4):35–84. 2.2

[Winograd, 1972] Winograd, T. (1972). *Understanding Natural Language*. NY Academic Press. 2.1

[Winograd, 1987] Winograd, T. (1987). Natural language: the continuing challenge. *AI Expert*, 2(5):7–8. 2.1

[Witten et al., 1999] Witten, I., Moffat, A., and Bell, T. (1999). *Managing Gigabytes: Compressing and Indexing documents and images (2nd edition)*. Morkan Kaufmann Publishers. 2.5

[Woods, 1977] Woods, W. (1977). Lunar rocks in natural English: Explorations in Natural Language Question Answering. In Zampolli, A., editor, *Linguistic Structures Processing*, volume 5 of *Fundamental Studies in Computer Science*, pages 521–569. North Holland. 2.1

[Yakushiji et al., 2001] Yakushiji, A., Tateisi, Y., and Miyao, Y. (2001). Event Extraction from Biomedical Papers Using a Full Parser. In *Proceedings of Pacific Symposium on Biocomputing*, pages 408–419, River Edge, N.J. World Scientific Publishing. 7.1.2.2

[Yakushiji et al., 2004] Yakushiji, A., Tateisi, Y., and nad Jun'ichi Tsujii, Y. M. (2004). Finding Anchor Verbs for Biomedical IE using Predicate-Argument Structures. In *Companion Volume to the Proceedings of 42st Annual Meeting of the ACL*, pages 157–160, Barcelona, Spain. 7.1.2.2

[Yang and Chua, 2003] Yang, H. and Chua, T.-S. (2003). Qualifier: Question answering by lexical fabric and external resources. In *EACL 2003*, pages 363–370. 2.3.4

[Zheng, 2003] Zheng, Z. (2003). Question answering using web news as knowledge base. In *The Eleventh Conference of the European Chapter of the Association for Computational Linguistics (EACL 2003). Budapest, Hungary. April 12-17, 2003*. 2.7

[Zweigenbaum, 2003] Zweigenbaum, P. (2003). Question answering in biomedicine. In *Proc. of EACL 03 Workshop: Natural Language Processing for Question Answering*, Budapest. 7

Employment History

1/01/00 - Present, IFI - University of Zurich (Switzerland), *Senior Researcher, Lecturer*

From 2000 to 2002 I worked in the WebExtrAns project, which (despite its name) was not concerned with Information Extraction from the Web, but rather with Answer Extraction from Technical Manuals. We collaborated with the University of Tartu (Estonia). From September 2002 to July 2005 I worked in the EU-funded project Parmenides, with many different European partners. In both WebExtrAns and Parmenides I acted as local coordinator. From June 2004 to March 2005 I had additionally a part-time position (20%) in the EU funded Network of Excellence REWERSE. Besides my activity in the projects, I have been responsible for GNU Software for the entire Institute (2001-2004) and webmaster of the CL group (2001-2002). Since March 2005 I have a permanent position. I am currently teaching a couple of classes per semester and doing research on Text Mining for Biomedical Literature within the context of the OntoGene project, of which I am local coordinator.

1/09/96 - 31/12/99, UMIST - Manchester (UK), *RA*

In this period I worked as a Research Assistant at UMIST. My activity was mainly related to the EU funded project FACILE (Information Extraction from Financial News) (4th Framework, Language Engineering, Project No. LE2440). The other participating institutions were Quinary SpA (Milan), Sema Group and Caja Segovia, (Spain), IRST (Trento) and Ofai (Vienna). FACILE was concerned with text categorization and related information extraction techniques applied to the selective dissemination of news to end users in the financial sector. During the project I coordinated our participation to the MUC7 Named Entity Extraction Task. MUC7 was the last of the Message Understanding Conferences. After the termination of the FACILE project I worked for a while on the EU funded project CONCERTO.

15/08/95 - 31/08/96, University of Udine & ITC/IRST (Italy), *junior RA*

In this period I had to remain available for a possible call to military service (which was then compulsory in Italy), so it was near impossible to find a real job, until I was finally discharged, without being called. In the period 15.08.1995 - 15.01.1996 I did some research work on User Modeling and on Information Retrieval within the AI group at the University of Udine. I also worked as an external consultant for IRST on User Modeling thematics. In the period 22.01.1996 to 31.08.1996 I worked as a junior Research Assistant at IRST. My work was related with the EU funded project GIST (Generation of Instructional Text)

15/03/95 - 15/08/95, GMD / IPSI - Darmstadt (Germany), *researcher*

From 15.03.95 to 14.06.95 I worked at GMD/IPSI (now Fraunhofer IPSI) as a Guest Researcher in the (former) KOMET group (Multilingual Text Generation). After my contract expired I stayed at IPSI two months longer on a voluntary basis. In this period I mainly worked at the on-line documentation of the "Generalized Upper Model" .

01/06/93 - 28/02/95, ITC / IRST - Trento (Italy), *student assistant*

I worked as student assistant in the (former) Natural Language Processing group of IRST (Istituto per la Ricerca Scientifica e Tecnologica - Trento/Italy) during the preparation of my master's thesis. My work dealt mainly with the Knowledge Base of the AlFresco prototype, however I had the opportunity to explore most of the aspects of the system. The work was supervised by Prof. Carlo Tasso (University of Udine) and by Dr. Oliviero Stock (IRST).

Computer Skills

- Programming Languages: in-depth knowledge of various languages (C, Lisp, Prolog, Java, Perl, etc.)

  I have a vast programming experience with C, Lisp, Prolog, Java and Perl, plus a fairly good knowledge of C++, Python, PHP, Javascript, Tcl/Tk and Pascal. I have some experience with a number of other programming languages (including FORTRAN, Basic, Modula II, OPS5, Smalltalk, Expect).

- Operating Systems: Unix/Linux at system administrator level

  I'm very familiar with Unix-based operating systems, including substantial elements of system administration knowledge (NIS, NFS, DNS). I have used various flavours of Linux, starting with Red Hat, moving then to Debian, and more recently to Ubuntu, which is now my favourite. I am proficient in unix shell scripting, in particular with bash. At IFI I was for a couple of years in charge of GNU software on the institute-wide Solaris system. I have good user-level knowledge of the Windows and Macintosh environments.

- Internet Protocols and Document Languages: I know well the most common Internet protocols (HTTP, FTP, NNTP, SMTP, POP, IMAP, MIME) and I am very familiar with HTML and XML-related technologies: XML, XSL, XSLT, CSS, DTD, XML Schema, XPath, SAX, DOM.

- Development Tools: I regularly use Open-Source tools, such as XEmacs, CVS, GCC, (GNU) make.

- Productivity Tools: Although familiar with the MS-Office family of products (Word, Powerpoint and Excel in particular), I prefer to use open-source tools, such as OpenOffice and LaTeX, when possible.

- Other: I have a good knowledge of Relational Databases (including the query language SQL , the ER model, and the - now defunct - Ingres DBMS).

Teaching

The courses which I have taught at the University of Zurich are listed below.

- XML: Fundamentals and Applications, Summer Semester 2005
- Question Answering Research Seminar, Winter Semester 2005/06
- Text Mining Research Seminar, Summer Semester 2006
- Applications of XML to CL, Summer Semester 2006
- Corpus Linguistics, Summer Semester 2007
- Text Mining, Summer Semester 2007
- XML: Fundamentals and Applications, Fall Semester 2007
- Corpus Linguistics, Spring Semester 2008
- Text Mining, Spring Semester 2008
- XML: Fundamentals and Applications, Fall Semester 2008
- Corpus Linguistics, Spring Semester 2009

Up to 2007 the academic year was divided into a summer and a winter semester. From then it has been divided into a fall and a spring semester.

Languages

- **Italian:** native
- **English:** fluent

  Since 1996 (when I moved to Manchester) I use English daily as a working language. My last official test of English was in February 1995, I took the TOEFL and obtained the mark of 643/670 (the way the marks are assigned has changed since then).

- **German:** advanced

  [R: near-native; U: advanced; S: good; W: medium] Since January 2000 I have lived in a German-speaking part of Switzerland and therefore have to use German for daily life. Over the years I attended numerous courses, and have passed the following examinations of the 'Goethe Institute' (the most authoritative center for the teaching of the German language): Zentrale Mittelstufenprüfung (2002), Kleines Deutsches Sprachdiplom (2003), Grosses Deutsches Sprachdiplom (2007).

- **Spanish:** advanced

  [R,U: near-native; S: good; W: medium]

- **French:** intermediate

  [R: good; U: medium; S: elementary; W: none]

- **Portuguese:** basic

  [R: good; U: basic; S: none; W: none]

- **Japanese:** elementary

  In 1998 and 1999 I attended Japanese courses at the Centre for Japanese Studies in Manchester. My final marks were 39/40 (reading), 38/40 (listening) and 17/20 (oral). In 2007 I attended an intensive one-month course in Tokyo to refresh my

knowledge of the language. Afterwards, I attended a number of courses at the language center of the University of Zurich. On Dec. 7th 2008 I took the official Japanese Language Proficiency Test (4), and passed it. At present, I can read and write hiragana and katakana. I know the meaning and pronunciation of about 100 Kanji characters. I can recognize the approximate meaning of about 400 Kanji characters. My conversational skills are rather limited.

R: ability to read written language, U: ability to understand spoken language, S: ability to speak the language, W: ability to write

Projects

This is a list of the projects that I have worked for, in various capacities, from research assistant to project manager

- ALFRESCO (1993-1995): a multimodal NLP system for a user interested in painters and frescoes of the 14th Century (an ITC-IRST internal project).
- LRE-GIST (1996): multilingual generation (English, German, Italian) of forms for public offices (funded by the European Union).
- LE-FACiLE (1996-99): Online News Classification (funded by the European Union).
- ESPRIT-Concerto (1999): development of improved techniques for indexing, searching and retrieving textual documents, by taking into account the meaning of these documents.
- WebExtrAns (2000-2002): Answer Extraction from technical manuals.

- Parmenides (2002-2004): Ontology driven Temporal Text mining on organizational data for extracting temporal valid knowledge. (local coordinator)
- REWERSE (2004)
- OntoGene (2005-present) (project manager)

Research funding

I am familiar with most of the EU funding instruments in FP7, as well as with Swiss national funding opportunities. I have obtained funding for the following projects:

2008 "A Comparitive Study of Syntactic Parsers and their Applications in Biomedical Text Mining" (funded by the Swiss National Science Foundation): I wrote the project proposal. SNF funding: 70,000 CHF. Duration: 18 months.

2007 "Detection of biological interactions from biomedical literature" (funded by the Swiss National Science Foundation): I wrote the project proposal. SNF funding: 114,046 CHF. Additional funding and support provided by Novartis Pharma AG. Duration: 18 months.

2003 "Parmenides" (EU FP5 project): I contributed to the project proposal, together with a number of European partners. Total Project Budget: 3,251,490 Euro. EU funding: 1,954,875 Euro. Swiss participants were funded separately. Funding of our group: 392,291 CHF (231,306 Euro). Duration: 36 months.

Academic Activities

Journal reviewer: ACM Computing Surveys, Artificial Intelligence in Medicine, BMC Bioinformatics, Computational Linguistics, Expert Systems, Language Resources and Evaluation, Natural Language Engineering, Journal of Biomedical Infomatics (Elsevier).

Scientific committee member (conferences and workshops): workshop on Ontologies and Information Extraction (2003), LREC 2004, QA in Restricted Domains (at ACL 2004), ROMAND2004 (at COLING2004), workshop on User Oriented Evaluation of Knowledge Discovery Systems (at LREC2004), workshop on the Application of Language and Semantic Technologies to support Knowledge Management Processes at EKAW 2004, special issue of the Journal Computer Speech and Language (Elsevier Science) on Multiword Expressions (2004), 3rd international workshop on Paraphrasing (IWP 2005), CICLING 2006, LREC 2006, ROMAND 2006, EACL 2006 (student session), CICLING 2007, ACL 2007 (poster and demo session), LREC 2008, ECCB 2008, NLDB 2008, CICLING 2009, NLDB 2009.

Reviewer (conferences, workshops and collections): New Directions in Question Answering edited by Mark Maybury (2003), Estudios de Lingüística Aplicada (special issue, 2003), Question Answering track at IJCNLP 2005, Computational Linguistics (special issue on Question Answering in Restricted Domains, 2005), 2nd international symposium on Languages in Biology and Medicine (LBM 2007), ACL HLT 2008 (Question Answering track), EMNLP 2008.

Conference organization committee member: COLING2004.

Selected Publications

I am author or co-author of 64 published papers, of which 60 peer-reviewed: 6 journal papers, 2 articles in books (excluding conference proceedings), 29 articles in international conferences, 19 in workshops. A complete and up-to-date list can be found at <http://www.cl.uzh.ch/CL/rinaldi/HOME/publications.html>.

Below a selection of five representative papers, which document some of the areas of my research activities (text mining [1,2], terminology and ontology learning [3], question answering [4], document annotation [5]).

1. Fabio Rinaldi, Thomas Kappeler, Kaarel Kaljurand, Gerold Schneider, Manfred Klenner, Simon Clematide, Michael Hess, Jean-Marc von Allmen, Pierre Parisot, Martin Romacker, Therese Vachon. OntoGene in BioCreative II. Genome Biology, 2008, 9:S13. doi:10.1186/gb-2008-9-s2-s13

2. Fabio Rinaldi, Gerold Schneider, Kaarel Kaljurand, Michael Hess, Martin Romacker. An environment for relation mining over richly annotated corpora: the case of GENIA. BMC Bioinformatics 2006, 7(Suppl 3):S3. doi:10.1186/1471-2105-7-S3-S3

3. Fabio Rinaldi, Elia Yuste, Gerold Schneider, Michael Hess, David Roussel. Exploiting Technical Terminology for Knowledge Management. In Ontology Learning from Text: Methods, Evaluation and Applications, Paul Buitelaar, Philipp Cimiano, Bernardo Magnini (editors). IOS Press, 2005.

4. Fabio Rinaldi, Michael Hess, James Dowdall, Diego Mollá, Rolf Schwitter. Question Answering in Terminology-rich Technical Domains. In "

New Directions in Question Answering" Maybury, M. T. editor. 2004. AAAI/MIT Press.

5. Fabio Rinaldi, James Dowdall, Michael Hess, Jeremy Ellman, Gian Piero Zarri, Andreas Persidis, Luc Bernard, Haralampos Karanikas. Multilayer annotations in Parmenides. The K-CAP2003 workshop on "Knowledge Markup and Semantic Annotation", October 25-26, 2003. Sanibel, Florida, USA.

Memberships and Affiliations

I am a member of the Swiss Informatics Society and the Swiss Group for Artificial Intelligence and Cognitive Science (SGAICO), which is a member of the European Coordinating Committee for Artificial Intelligence (ECCAI). Formerly, I was a member of the Italian Association for Artificial Intelligence. I am also a member of the Association for Computational Linguistics (ACL) and of the ACM.

Awards, Invited Presentations and Academic Visits

In 1991 I received an Erasmus grant from the European Community, thanks to which I attended some courses at the University of Nottingham, UK.
In 1994 I received a grant from DAAD (Deutscher Akademischer Austauschdienst – German Academic Exchange Service) thanks to which I attended the "Internationalen Sommerkurs" at the University of Mannheim in September 1994.
In 2008 I was awarded a research fellowship from SNF (Swiss National Science Foundation) in order to pursue independent research at NaCTeM

(National Center for Text Mining, Manchester, UK) and Tsujii Laboratories (University of Tokyo, Japan).

July 2007. Guest Researcher, Tsujii laboratory, University of Tokyo, Japan.

December 6th, 2003. Invited presentation at the workshop "Ontology Learning and Question Answering Tools from/for Large Text Collections", DaimlerChrysler Research Center, Ulm, Germany.

December 9th, 2005. Invited presentation at the Computational Knowledge Management and Text Mining Unit of Novartis, Basel, Switzerland.

July 7th, 2007. Invited presentation at Tsujii laboratory, University of Tokyo, Japan.

July 24th, 2008. Invited presentation at the HLT group, Fondazione Bruno Kessler, Trento, Italy.

November 7th, 2008. Invited presentation at the DBTA workshop on Information Retrieval (Algorithms and Systems for Text and Multimedia Retrieval), Basel, Switzerland.

June 8th, 2009. Invited presentation at the Center for Computational Pharmacology - University of Denver, Colorado.

July 6th, 2009. Invited presentation at NaCTeM - The National Center for Text Mining, Manchester, UK.

July 8th, 2009. Invited presentation at EBI - European Center for Bioinformatics, Hinxton, Cambridge, UK.